VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

AUTOMATICKÉ STROJOVÉ METODY ZÍSKÁVÁNÍ ZNALOSTÍ Z MULTIMEDIÁLNÍCH DAT
AUTOMATIC MACHINE LEARNING METHODS FOR MULTIMEDIA DATA ANALYSIS

DIZERTAČNÍ PRÁCE
DOCTORAL THESIS

AUTOR PRÁCE — **Ing. Jan Mašek**
AUTHOR

ŠKOLITEL — **doc. Ing. Radim Burget, Ph.D.**
SUPERVISOR

BRNO 2016

ABSTRAKT

Kvalitní a efektivní zpracování rostoucího množství multimediálních dat začíná být v dnešní době stále více potřebné pro získání určité znalosti z těchto dat. Práce se zabývá výzkumem, implementací, optimalizací a experimentálním ověřením automatických metod strojového učení pro získávání znalostí z multimediálních dat, kde bylo v řadě příkladů dosaženo vyšší přesnosti ve srovnání s konvenčními metodami a vybrané výsledky byly publikovány v časopisech s impaktním faktorem [1, 2]. K tomu byly v práci speciálně vytvořeny výpočetní metody, které využívají masivně paralelní hardware, díky kterému je dosaženo úspory elektrické energie a výpočetního času při dosažení lepší přesnosti řešených problémů. Výpočty trvající běžně v řádech dní bylo možné urychlit novými metodami na několik málo minut. Funkčnost vytvořených metod byla ověřena na vybraných problémech: detekce krční arterie z ultrazvukových snímků a následné určení stupně nemoci této arterie, detekce staveb z leteckých snímků pro získání jejich zeměpisných souřadnic, detekce jednotlivých materiálů obsažených v meteoritu ze snímků počítačové tomografie, zpracování velkých databází strukturovaných dat, klasifikace hutních materiálů s pomocí laserové spektrometrie a automatická klasifikace emocí z textů.

KLÍČOVÁ SLOVA

Akcelerace výpočtů, detekce objektů, GPU, hluboké učení, masivně paralelní hardware, multimediální data, paralelní zpracování, trénovatelná segmentace, umělá inteligence, úspora energie.

ABSTRACT

The quality and efficient processing of increasing amount of multimedia data is nowadays becoming increasingly needed to obtain some knowledge of this data. The thesis deals with a research, implementation, optimization and the experimental verification of automatic machine learning methods for multimedia data analysis. Created approach achieves higher accuracy in comparison with common methods, when applied on selected examples. Selected results were published in journals with impact factor [1, 2]. For these reasons special parallel computing methods were created in this work. These methods use massively parallel hardware to save electric energy and computing time and for achieving better result while solving problems. Computations which usually take days can be computed in minutes using new optimized methods. The functionality of created methods was verified on selected problems: artery detection from ultrasound images with further classifying of artery disease, the buildings detection from aerial images for obtaining geographical coordinates, the detection of materials contained in meteorite from CT images, the processing of huge databases of structured data, the classification of metallurgical materials with using laser induced breakdown spectroscopy and the automatic classification of emotions from texts.

KEYWORDS

Computation acceleration, object detection, GPU, deep learning, massively parallel hardware, multimedia data, parallel processing, trainable segmentation, artificial intelligence, energy saving.

MAŠEK, Jan *Automatické strojové metody získávání znalostí z multimediálních dat*: dizertační práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2016. 117 s. Vedoucí práce byl doc. Ing. Radim Burget, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou doktorskou práci na téma „Automatické strojové metody získávání znalostí z multimediálních dat“ jsem vypracoval(a) samostatně pod vedením vedoucího doktorské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor(ka) uvedené doktorské práce dále prohlašuji, že v souvislosti s vytvořením této doktorské práce jsem neporušil(a) autorská práva třetích osob, zejména jsem nezasáhl(a) nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom(a) následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora(-ky)

Zde bych rád poděkoval svému školiteli disertační práce doc. Ing. Radimovi Burgetovi, Ph.D. za pomoc, cenné rady, jeho trpělivost a čas při zpracování mé disertační práce.
V neposlední řadě děkuji své rodině za podporu během mého studia.

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

PODĚKOVÁNÍ

Výzkum popsaný v této doktorské práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

Brno . . . . . .

. . . . . .

podpis autora(-ky)

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OP Výzkum a vývoj
pro inovace

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

V poslední době dochází k nárůstu množství digitálních dat, ze kterých lze jejich vhodným zpracováním získat určitou znalost. Důraz je kladen nejen na kvalitu zpracování dat, ale i na rychlost, s jakou jsou tyto operace prováděny. Proto je nezbytné na jedné straně vyvíjet nové algoritmy a metody, které budou dosahovat lepší přesnosti při zpracování dat, ale zároveň se zabývat i optimalizací stávajících metod pro nasazení na speciálních typech výpočetního hardwaru, který výrazně urychlí výpočty. Tyto optimalizované metody dokáží snížit čas výpočtu trvající běžně dny na několik málo minut a zároveň díky speciálním typům hardware dochází k výraznému snížení spotřeby elektrické energie.

Automatické metody strojového učení mohou být nasazeny na celou řadu dnešních problémů, mezi které například patří dolování informací z obrazu, textu, zvuku nebo videa. Příkladem může být detekování objektů z medicínských snímků, klasifikace satelitních dat, kategorizace obrazů, detekce objektů z fotoaparátů a kamer. Tato data většinou obsahují určité množství nepodstatných informací, které jsou metodami pro dolování informací z báze dat odfiltrovány a po další analýze je použito jen to podstatné.

Dolování znalostí z dat je v mnoha směrech náročná úloha a jejím problémem je hlavně nedostatečná přesnost detekce nebo klasifikace v důsledku zašumění dat. Dalším problémem je výpočetní náročnost mnohých algoritmů strojového učení a tím vznikající potřeba jejich akcelerace. Konkrétním příkladem může být detekce krční arterie z ultrazvukových snímků a následné určení stupně nemoci této arterie, nebo detekce domů z leteckých snímků pro získání jejich zeměpisných souřadnic, či detekce jednotlivých materiálů meteoritu ze snímků počítačové tomografie, případně zpracování velkých databází strukturovaných dat. Tyto příklady potřebují být řešeny s co nejvyšší přesností nejlépe v reálném čase.

Hlavním přínosem práce je výzkum, implementace, experimentální ověření a optimalizace automatizovaných strojových metod pro získávání znalostí z multimediálních dat, kde bylo na vybraných problémech dosaženo vyšší přesnosti ve srovnání s konvenčními metodami. Vybrané výsledky byly publikovány na mezinárodních konferencích a také v impaktovaných časopisech [1, 2]. V rámci práce se podařilo vybrané algoritmy akcelerovat, čímž bylo dosaženo úspory elektrické energie i výpočetního času při dosažení přesnějších výsledků. Tyto metody je možné použít a kombinovat v rámci jednoho komplexního systému.

Funkčnost vytvořených metod byla ověřena na vybraných problémech: při detekci arterií z ultrazvukových snímků s pomocí evolučně optimalizovaného objektového detektoru [1] bylo při zpracování v reálném čase dosaženo 96 % úspěšnosti detekce pro data zdravých pacientů a 95 % úspěšnosti pro data nemocných pacientů. U dosa-

vadních metod nepřesáhla přesnost jejich detekce 90 % a zároveň proces neprobíhal v reálném čase. Pro určení stupně nemoci artérie byla provedena následná klasifikace s pomocí algoritmů hlubokého učení, kde bylo dosaženo 98 % přesnosti klasifikace pro 3 úrovně nemoci. Další navržená metoda [3, 4], která vylepšila současný stav problematiky, byla optimalizována pro speciální hardware, kde bylo díky této optimalizaci dosaženo pro algoritmus k-nejbližších sousedů 882-ti násobné zrychlení v porovnání s běžnou verzí algoritmu. Jiné optimalizované přístupy dosáhly maximálně 336-ti násobné zrychlení. Vytvořený optimalizovaný algoritmus pro trénování objektového detektoru dosáhl 137-mi násobného zrychlení.

Klasifikace hutních materiálů pomocí laserové spektrometrie [5] za použití nově navržených postupů pro zpracování obrazu dokázala oproti běžným postupům výrazně zvýšit celkovou přesnost na 99,1 % při klasifikaci materiálů do 50-ti tříd. Dále byla ověřena přesnost klasifikace při automatickém rozpoznávání emocí z textu. Výsledky byly publikovány v impaktovaném časopise [2] (IF=0,59). Dosažená přesnost byla 86,89 % při klasifikaci do pěti emočních tříd, přičemž bylo dosaženo 11,4 % zlepšení oproti současným přístupům. Dále byla funkčnost metod ověřena na příkladech detekce domů ze satelitních snímků s dosaženou přesností detekce 78 %. Stejnou metodou byly z obrazů počítačové tomografie detekovány jednotlivé materiály obsažené v meteoritech, kde výsledkem byly získané objemy těchto obsažených materiálů.

Práce je členěna následovně: Kapitola 1 rozebírá současné přístupy v oblastech detekce objektů v obrazech, učících se algoritmů umělé inteligence, dále popisuje problematiku velmi nového přístupu hlubokého učení a nakonec je uvedena metoda trénovatelné segmentace. Kapitola 2 vytyčuje cíle disertace, kterými jsou zejména vývoj optimalizovaných algoritmů, vytvoření systému pro trénování a testování objektového detektoru, vytvoření nových databází dat a otestování vytvořených algoritmů na těchto databázích. Vlastní stav řešení je popsán v kapitole 3. Zde byl navržen systém pro detekci objektů v obrazech, rychlostní optimalizace jeho výpočetně náročných funkcí, dále evoluční optimalizace výstupního objektového detektoru a zakomponování algoritmů hlubokého učení do systému pro klasifikaci detekovaných objektů. V této kapitole byla dále řešena optimalizace algoritmu k–nejbližších sousedů a rychlostní optimalizace metody trénovatelné segmentace. Kapitola 4 uvádí výsledky, které byly dosaženy během provedených experimentů s vytvořenými algoritmy. Jedná se o detekci domů ze satelitních snímků, segmentaci snímků meteoritů, testování optimalizovaných verzí algoritmů k–nejbližších sousedů a algoritmu pro trénování objektového detektoru. Kapitola je zakončena výsledky týkajících se přesností detekce arterie a její klasifikace podle stupně nemoci. V kapitole 5 jsou diskutovány výhody a nevýhody vytvořeného systému a dosažené výsledky jsou srovnány s jinými současnými pracemi. Práce je zakončena vyhodnocením v kapitole 6.

1 DOSAVADNÍ STAV VÝVOJE

Výzkum v rámci této doktorské práce se zabýval návrhem a optimalizací metod strojového učení pro získávání znalostí z obrazových dat. Zvolené metody strojového učení byly vytvořeny za účelem zpřesnit řešení vybraných problémů a pro výpočetně náročné operace byly vytvořeny optimalizované verze algoritmů schopné díky distribuovaným systémům provádět celkovou analýzu výrazně efektivněji.

Výzkum byl konkrétně zaměřen na metody týkající se detekce objektů v obrazech, na výkonnostní optimalizaci metod strojového učení a na řešení problémů s pomocí metody trénované segmentace [6].

Navržený systém pro detekci objektů v obrazech vychází z původní metody pánů Violy a Jonese [7], která byla rozšířena o možnosti použití dalších obrazových příznaků, které za použití genetických algoritmů a metod hlubokého učení, zvýšily přesnost při řešení problému detekce arterie z ultrazvukových snímků [1]. Vybrané části tohoto systému byly optimalizovány pro možnost spouštění v distribuovaných systémech za účelem snížení výpočetního času.

Další navržený systém je určen pro segmentaci obrazových dat s pomocí učících se algoritmů umělé inteligence. Zde byly, za pomoci metody trénovatelné segmentace ve spojení s běžnými metodami pro zpracování obrazu, navrženy procesy pro řešení vybraných problémů [8, 6, 9].

Následující podkapitoly popisují teorii, která se váže k výše zmíněním navrženým systémům.

1.1 Detekce objektů v obrazech

Detekce objektů v obrazech je již řadu let stále se rozvíjející směr, kde ještě mnoho problémů nebylo zcela vyřešeno. Příkladem nohou být práce týkající se detekce osob v obrazu [10], kde systémy pro detekci osob mohou být součástí dopravních prostředků [11], které jsou schopné včas detekovat chodce na vozovce a patřičně zareagovat [12]. Samy dopravní prostředky mohou byt detekovány kamerovými systémy za účelem určení plynulosti provozu [13, 14] nebo pro určení typů a počtů dopravních prostředků. Jedním dnes velmi diskutovaným tématem je detekce a klasifikace většího počtu různých objektů v obrazech [15, 16].

Tato část se zabývá popisem současných metod pro detekci objektů v obrazech. Jsou zde popsány nejvíce používané metody, na základě kterých byl v další části práce vytvořen propracovaný automatický systém pro detekci objektů v obrazech. Funkčnost tohoto systému byla ověřena zejména na vybraných příkladech z oblasti zdravotnictví.

1.1.1 Metoda Viola-Jones

Jedna z hlavních metod pro detekci objektů v obrazech byla uvedena P. Violou a M. Jonesem v roce 2001 [7]. Tento detektor objektů v obrazech byl původně určen pro detekci obličejů a je označován jako Viola-Jones detektor. Detektor je v dnešní době velmi známý a má užití v celé řadě problémů. Jeho vlastností je zejména rychlost, škálovatelnost a dobrá přesnost detekce. Detektor je trénován algoritmem AdaBoost [17], který používá Haarovy příznaky [18] jako slabé klasifikátory. Vstupem algoritmu AdaBoost jsou positivní a negativní snímky. Rychlá funkce detektoru byla dosažena použitím integrálního obrazu a kaskády klasifikátorů.

Zmíněný přístup byl použit v mnoha jiných pracích. V [19] je popsána metoda trénování obličejového detektoru, kde detekované obličeje mají různý úhel natočení. Dále je metoda schopna detekovat obličeje jak zepředu, tak i z boku. Článek Violy a Jonese [20] popisuje detekci natočených obličejů za použití rozšířené sady Haarových příznaků. V [21] je trénován obličejový detektor, který zároveň zobrazuje hodnotu úhlu, o kterou byl obličej natočen. Použití evolučních algoritmů pro snížení počtu klasifikátorů k detekování natočených obličejů bylo popsáno v [22]. Detekce zvířecích hlav byly popsány v článcích [23] a [24]. V článku [25] byl natrénován detektor pro detekci lidské ruky. Dále byla metoda Viola-Jones použita v práci [26] zabývající se detekcí a klasifikací dospělých osob z obrazů, dále v monitorovacích bezpečnostních systémech [27] nebo v [28] bylo detekováno zívání řidiče dopravního prostředku.

Jiná část prací se zabývá optimalizací metody Viola–Jones pro běh na masivně paralelním hardware jako jsou například grafické akcelerátory (GPU Graphic Processing Unit). Práce zaměřené na výpočetně optimalizovaný obličejový detektor se zabývaly zrychlením detekční části na GPU s použitím paralelní technologie CUDA (Compute Unified Device Architecture) [29]. Obličeje byly detekovány z videa o vysokém rozlišení v [30]. V [31] byl popsán rychlý systém pro detekování a sledování obličeje ve videu. Akcelerovaná verze detektoru založená na knihovně pro zpracování obrazu OpenCV[1] (Open Source Computer Vision) je popsána v [32]. Implementace detektoru vytvořená na paralelní platformě OpenCL[2] (Open Computing Language) je popsána v [33]. V [34] byla vytvořena OpenCV implementace detektoru, která byla nasazena v SoC (System–on–Chip) prostředích.

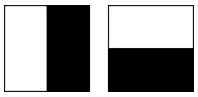
Modifikace Viola–Jones detektoru byly použity autorem této práce pro řešení různých problémů: [35, 1, 36, 37, 38]. Výsledky některých těchto prací budou podrobněji popsány v kapitole 4.

[1] Dostupné z URL: `http://opencv.org/`

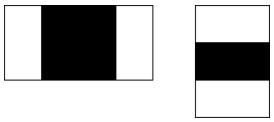
[2] Dostupné z URL: `http://www.khronos.org/opencl/`

Haarovy příznaky

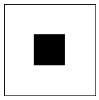
Metoda Viola-Jones je, jak již bylo řečeno, primárně založena na Haarových příznacích. Tento příznak se skládá ze dvou obdélníků, černého a bílého. Hodnota odezvy Haarova příznaku na vstupní snímek je pak spočtena jako rozdíl součtu hodnot v bílé a černé oblasti. Příznaky se dělí na hranové (viz obrázek 1.1), čárové (viz obrázek 1.2) a středové (viz obrázek 1.3). Pro trénovací okno o velikosti 24x24 pixelů (použito pro obličejový detektor) je generováno například 189 664 takovýchto příznaků.



Obr. 1.1: Hranové příznaky.



Obr. 1.2: Čárové příznaky.



Obr. 1.3: Středový příznak.

Integrální obraz

Výpočet odezvy Haarova příznaku je časově náročný proces, proto byl zaveden integrální obraz, který výrazně urychlí tento výpočet. Integrální obraz je spočten ze vstupního obrazu tak, že každý pixel integrálního obrazu je roven sumě všech pixelů nacházejících se nad tímto pixelem a vlevo od pixelu [40]. Proces je vyjádřen rovnicí 1.1:

$$i_{\text{I}}\,(u, v\,) = \sum_{u\acute{}\leq u, v\acute{}\leq v} i(\,u\acute{}, v\acute{}), \tag{1.1}$$

kde $i_{\text{I}}\,(u, v)$ jsou hodnoty integrálního obrazu a $i(\,u\acute{}, v\acute{})$ jsou hodnoty pixelů vstupního obrazu. Vstupní obraz je převeden na integrální s pomocí rovnic:

$$s(u,v) = s(u,v-1) + i(u,v), \tag{1.2}$$

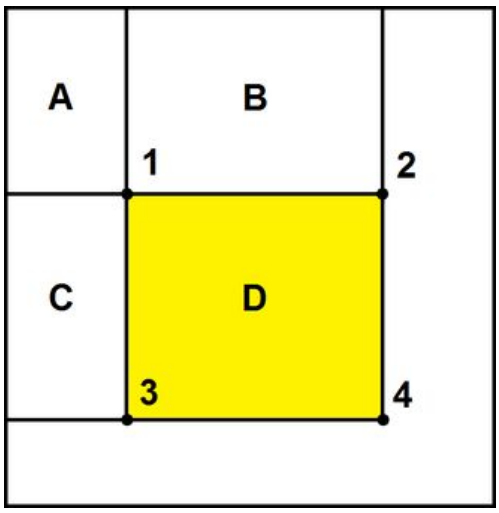
$$i_{\mathrm{I}}(u,v) = i_{\mathrm{I}}(u-1,v) + s(u,v), \tag{1.3}$$

kde $s(u,v)$ je součet všech hodnot pixelů v řádku, při dodržení podmínek $s(u,-1) = 0$ a $i_{\mathrm{I}}(-1,v) = 0$. Pro výpočet žlutě označené oblasti na obrázku 1.4 je třeba spočítat hodnoty v jednotlivých bodech 1 až 4. Například hodnota v bodě 3 je spočtena jako součet hodnot pixelů definovaných oblastmi A a C. Hledaná žlutá oblast D je tedy spočtena jako $4 - 3 - 2 + 1$. [7]

Kvadrát hodnot integrálního obrazu i_{ISqr} slouží pro následný výpočet normalizované odezvy Haarova příznaku, kde i_{ISqr} je spočten stejným způsobem jako již uvedený integrální obraz poouze s tím rozdílem, že $i(u,v)$ je nahrazeno $i^2(u,v)$. Z důvodů rozdílné intenzity osvětlení pro různé trénovací snímky je třeba provést normalizaci odezvy Haarova příznaku. K tomu je použit vzorec [40]:

$$f(x)' = \frac{f(x)}{wh\sqrt[2]{\frac{i_{\mathrm{ISqr}}(w,h) - \frac{{i_{\mathrm{I}}}^2(w,h)}{wh}}{wh}}}, \tag{1.4}$$

kde $f(x)'$ je normalizovaná odezva, $f(x)$ je původní odezva Haarova příznaku na snímek x, w je šířka snímku, h je výška snímku, $i_{\mathrm{I}}(w,h)$ je součet hodnot všech pixelů vstupního snímku o velikosti wh, $i_{\mathrm{ISqr}}(w,h)$ je součet kvadrátů všech hodnot pixelů snímku.



Obr. 1.4: Příklad výpočtu sumy v obdélníkové oblasti integrálního obrazu.

Lineární klasifikátor

Lineární klasifikátor je označován jako slabý, protože jeho účelem je zařadit data do klasifikačních tříd s přesností větší než 50 % (bráno pro 2 klasifikační třídy – v tomto případě pro positivní a negativní). Tento klasifikátor klasifikuje snímek x do positivní nebo negativní třídy podle rovnice:

$$h(x, f, p, \Theta) = \begin{cases} 1 & pf(x) < p\Theta, \\ 0 & \text{jinak} \end{cases} \tag{1.5}$$

kde f je odezva Haarova příznaku na snímek x, p značí kladnou nebo zápornou polaritu a Θ značí prahovou hodnotu lineárního klasifikátoru. Pro určení klasifikační třídy je nejprve spočtena odezva příznaku na snímek x. Jestli je positivní klasifikační třída větší než prahová hodnota tak $p = 1$, jinak $p = -1$. Při konečném porovnání odezvy příznaku s prahovou hodnotou Θ je rozhodnuto, jestli snímek x je klasifikován jako positivní $h(x) = 1$ nebo jako negativní $h(x) = 0$. [41]

Kaskáda klasifikátorů

Kaskáda klasifikátorů slouží k výraznému urychlení detekčního procesu. Skládá se z určitého počtu stupňů, kde každý stupeň obsahuje určité množství slabých klasifikátorů. Jeden stupeň kaskády je v podstatě silný monolitický nelineární klasifikátor $H(x)$ s prahovou hodnotou P, podle které je aktuální pod okno (výřez velkého snímku, který je klasifikován) označeno za positivní nebo negativní. Tento silný klasifikátor je popsán rovnicí:

$$H(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq P, \\ 0 & \text{jinak} \end{cases} \tag{1.6}$$

kde α_t je váha slabého klasifikátoru h_t. Účelem této kaskády klasifikátorů je zamítnout na každém stupni co nejvíce negativních pod oken a poslat na následující stupeň ty positivní.

Trénování

Pro naučení detektoru, jaké objekty má detekovat, musí být provedeno jeho trénování. Proces začíná načtením vstupních trénovacích positivních snímků (obsahující hledaný objekt) a negativních snímků (obsahujících pozadí) a jejich převedením do integrálního obrazu. Spolu s vygenerovanými Haarovými příznaky jsou vstupem algoritmu AdaBoost, který vybere řádově stovky slabých klasifikátorů, které složí do

kaskády klasifikátorů. Tato kaskáda je pak výstupem trénovacího procesu a je uložena jako natrénovaná znalost (detektor). Toto trénovací schéma bude obdobné pro další typy příznaků probraných v kapitolách 1.1.2 a 1.1.3.

Detekce

Aplikováním detektoru na neznámý snímek obsahující rozličné objekty začne vyhledávání definovaného objektu, na který byl detektor trénován. V detekční fázi je načtena kaskáda klasifikátorů spolu se vstupními obrázky, ve kterých budou hledány objekty (jedná se o velké obrázky, které kromě hledaného objektu obsahují také pozadí). Objekty jsou pak hledány v každém obrázku následovně: nejprve je v obrázku nastavena počáteční velikost a pozice detekčního pod okna. Dále je rozhodnuto (aplikováním kaskády klasifikátorů), jestli pod okno obsahuje hledaný objekt. Poté je pod okno posunováno o určitý krok, dokud neprojde přes celý snímek. V následující iteraci je zvětšeno a znovu posouváno. Rozměry jsou tímto způsobem zvětšovány až do okamžiku, kdy jsou rozměry pod okna větší než rozměry vstupního snímku. Všechny nalezené detekce jsou dále sloučeny, zpracovány a zobrazeny.

1.1.2 Metoda histogramu orientovaných gradientů

Metoda histogramu orientovaných gradientů (HOG – Histogram of Oriented Gradients) [42] patří spolu s metodou Viola–Jones mezi nejpoužívanější metody pro detekci objektů v obrazech. Prvotně byla používána při trénování detektoru pro detekci osob. Principem metody je nalézt a spočítat výskyty orientovaných gradientů hran v definované positivní a negativní části obrazu a na základě takto získaných příznaků natrénovat detektor s pomocí učících se algoritmů umělé inteligence (neuronová síť, rozhodovací strom, ...).

Metoda HOG byla základem mnoha prací: v [42] je popsáno podrobné nastavení parametrů této metody. Detekce lidské postavy po částech je popsána v [43], kde bylo dosaženo 98 % přesnosti. Pro detekci obličeje byla metoda použita v práci [44]. Navržený systém je založen na kaskádě klasifikátorů a dosáhl 70-ti násobného urychlení oproti systému, ze kterého vycházel. Práce [45] vychází z metody Viola–Jones, kde hlavním rozdílem je použití HOG příznaků místo Haarových příznaků. V [46] je použito více typů deskriptorů kombinovaných s příznaky HOG. Navržený systém byl nasazen na detekci objektů z videa. V [47] je popsána detekce chodců z denních a nočních snímků (za použití infračervených kamer). Pro snímání byla použita stereo kamera. Práce [48] popisuje komplexní systém pro detekci chodců. V [49] je metoda HOG použita pro detekování dopravních prostředků z leteckých snímků a využití metody v systému pro detekci dopravních značek je popsáno v [50].

Výpočet příznaku

Výpočet příznaků HOG začíná zjištěním gradientu pro každý pixel v obraze. Gradient [51] je definován jako vektor ∇f parciálních derivací pro osy x a y:

$$\nabla f(x, y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right). \tag{1.7}$$

Například pro výpočet gradientu $\frac{\partial f}{\partial x}$ ve směru x je použit jednorozměrný filtr $[-1, 0, 1]$, kterým je na vstupní obrazová data f aplikována operace konvoluce:

$$f' \approx [-1, 0, 1] * f, \tag{1.8}$$

kde f' je konvolvovaný výstup. Dále je pro každý bod vstupního obrazu spočten modul gradientu (magnituda) podle rovnice:

$$\|\nabla f(x, y)\| = \sqrt{f'_x(x, y)^2 + f'_y(x, y)^2} \tag{1.9}$$

a úhel gradientu podle rovnice:

$$\psi = \arctan \frac{f'_x(x, y)}{f'_y(x, y)}, \tag{1.10}$$

kde x a y jsou pozice pixelu v obrázku. Poté se gradienty rozdělí podle úhlů do definovaného počtu skupin. A následně jsou vytvořeny histogramy pro jednotlivé buňky. Buňky jsou dále seskupeny do bloků, je provedena normalizace a jsou vytvořeny výstupní deskriptory. Takto vygenerované hodnoty jsou použity při trénování modelu umělou inteligencí.

Podle metody HOG byly vytvořeny obrazové příznaky s různými modifikacemi. Vše bude podrobněji popsáno v další části práce.

1.1.3 Metoda lokálních binárních vzorů

Metoda lokálních binárních vzorů (LBP - Local Binary Patterns) je běžně používána pro zpracování obrazových dat obsahujících texturu [52]. Výhodou této metody je robustnost vůči osvětlení, invariance vůči rotaci a výpočetní jednoduchost. Tato metoda byla prvotně používána pro klasifikaci textur, ale později byla využita i pro detekci objektů. Současné přístupy se například zabývají problematikou detekcí emocí z obličeje [53] nebo systémem pro automatické detekování zvířat dopravními prostředky [54].

Výpočet vzoru

Výpočet vzoru LBP je na principu porovnání hodnot jasů středového pixelu s jeho okolím (8-mi okolím pro původní verzi metody), kdy hodnoty menší než je hodnota tohoto středového pixelu jsou nastaveny na 0, zbytek na 1. Následně je spočítán identifikátor LBP vzoru (viz obrázek 1.5). Výpočet LBP vzoru v bodě obrazu (x_c, y_c) je dán rovnicí [55]:

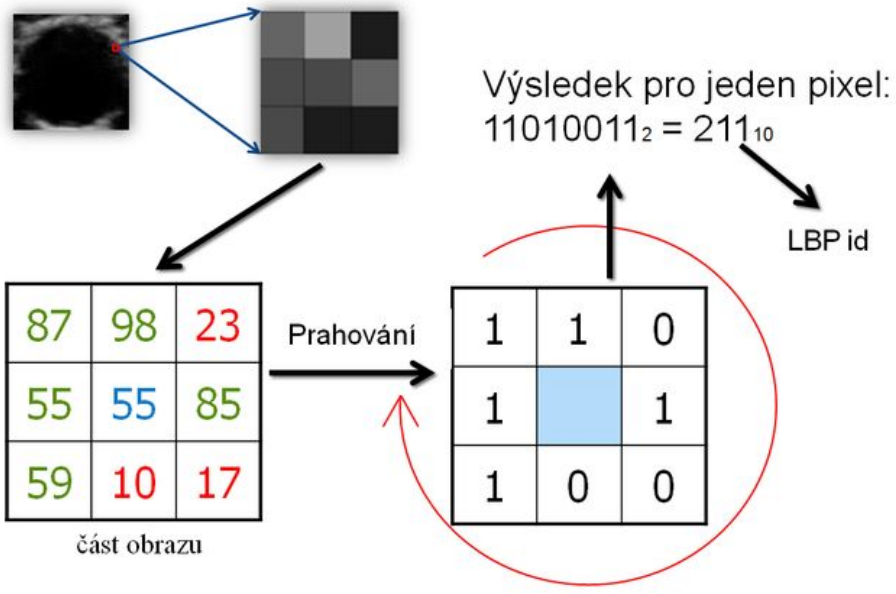
$$LBP\left(x_c, y_c\right) = \sum_{n=0}^{N} s\left(I_n - I_c\right) 2^n, \tag{1.11}$$

kde I_c je hodnota středního pixelu, I_n je hodnota sousedního pixelu, $N = 8$ určuje 8-mi bitový rozměr LBP vzoru. Pro výpočet funkce s při tom platí následující rovnice:

$$s(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0, \end{cases} \tag{1.12}$$

kde x je hodnota rozdílu $I_n - I_c$ z předcházející rovnice.

Takto jsou spočítány vzory pro každý pixel v obraze a následně je z těchto vzorů sestaven histogram, který je dále použit učícími se algoritmy umělé inteligence. Tyto LBP vzory jsou v další části práce použity jako základ vytvořených LBP příznaků.



Obr. 1.5: Ukázka výpočtu LBP.

1.2 Učící se algoritmy umělé inteligence

Učící se algoritmy umělé inteligence patří do metod strojového učení a slouží pro analýzu strukturovaných dat (tzv. data mining). Multimediální nestrukturovaná data, jakými jsou text [56, 2], audio [57], video sekvence a statické obrazy, musí být před aplikováním učících se algoritmů předzpracovány [58, 59, 60]. Toho je dosaženo zvolením a vygenerováním vhodných příznaků, které po aplikování na nestrukturovaná data extrahují hodnoty vhodné pro učící se algoritmy [5]. Tato práce se bude zabývat pouze učícími se algoritmy s tzv. učitelem. Principem těchto algoritmů je na základě vstupního ohodnoceného souboru dat natrénovat znalost (trénovací proces – obvykle bývá výpočetně náročný) a poté aplikovat tuto znalost (testovací proces – výpočetně nenáročný) na neohodnocených datech s cílem tato data klasifikovat s co nejlepší přesností.

Mezi nejvíce používané učící se algoritmy patří [61]:

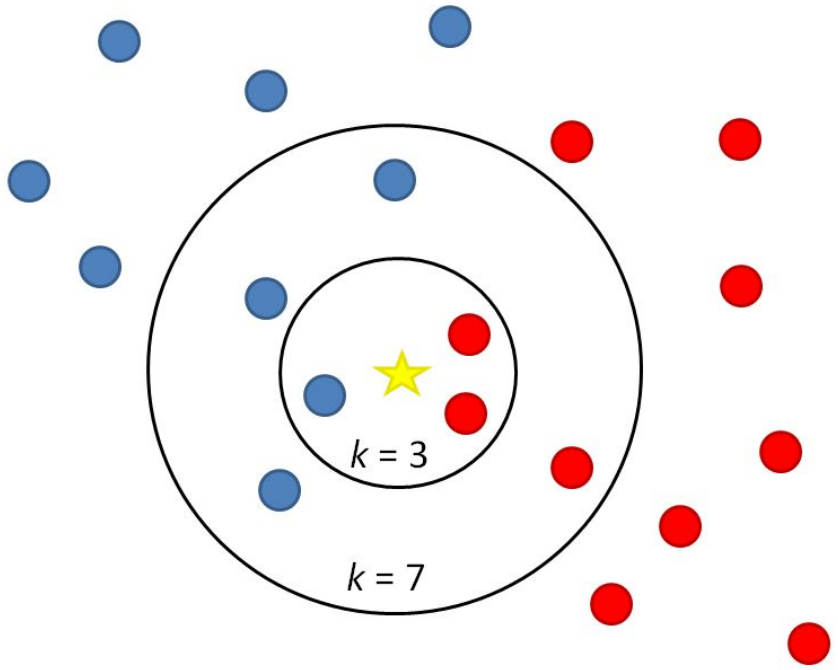
- k–nejbližších sousedů (k–NN - k–Nearest Neighbors),
- systémy podpůrných vektorů (SVM - Support Vector Machines),
- neuronové sítě (NN - Neural Networks),
- rozhodovací stromy (DT - Decision Trees),
- náhodné lesy (RF - Random Forests).

V rámci tématu této práce budou některé s těchto algoritmů použity v navrženém systému pro detekci objektů v obrazech nebo spolu s algoritmem trénovatelné segmentace. Protože s rostoucím množstvím multimediálních dat vzniká potřeba tato data zpracovávat v čase akceptovatelném pro danou aplikaci a do současné doby nebyl navržen vhodný algoritmus, který by byl schopen urychlit výpočet trvající například týdny na několik hodin, byl v rámci práce jako vhodný kandidát pro optimalizaci na GPU zvolen algoritmus k–nejbližších sousedů.

1.2.1 Algoritmus k–nejbližších sousedů

Algoritmus k–nejbližších sousedů (k–NN - k–Nearest Neighbors) patří mezi základní a velmi používané algoritmy umělé inteligence. Algoritmus je používán pro klasifikaci i pro regresi. Princip algoritmu je zobrazen na obrázku 1.6. V trénovací fázi jsou do modelu uloženy informace o hodnotách trénovacích prvků a při testování je pro každý testovací prvek spočtena vzdálenost (Euklidovská, Manhattanská, ...) mezi atributy tohoto prvku s atributy trénovacího prvku. Tímto způsobem jsou spočteny vzdálenosti se všemi trénovacími vzorky. Vzdálenosti jsou pak seřazeny a k vzorků

s nejnižšími vzdálenostmi jsou zvoleny jako nejbližší sousedé. Podle klasifikačních tříd k nejbližších trénovacích prvků je testovací prvek klasifikován.



Obr. 1.6: Princip algoritmu k-NN.

1.2.2 Optimalizované verze algoritmu *k*–NN

Některé články popisují jiné přístupy a využití optimalizované verze k–NN algoritmu. V [62] je srovnána implementace GPU verze k–NN s několika CPU verzemi. V [63] a [64] bylo implementováno několik speciálních technik pro maximalizaci využití grafického akcelerátoru. k-NN algoritmus pro analýzu textury je popsán v [65]. V [66] byl představen LSH (Locality Sensitive Hashing). Výsledky byly ověřeny na velkých obrazových databázích s dosaženou 40-ti násobnou akcelerací. Implementace k–NN algoritmu do prostředí RapidMineru [67] je popsána v [68]. Algoritmus dosahuje 150-ti násobného zrychlení, ale dokáže zpracovat databáze s maximálně 128 atributy.

Původní metoda vytvořená v rámci této disertační práce byla již publikována v článcích [4, 3] a dosahuje 882-ti násobného zrychlení a počet atributů a dat, které dokáže zpracovat, je omezen pouze pamětí grafického akcelerátoru. Podrobnosti o metodě budou uvedeny v další části práce.

1.3 Hluboké učení

Hluboké učení (DL – Deep Learning) je v posledních letech velmi používaný přístup pro řešení složitějších problémů zejména v oblasti klasifikace obrazů. Rozdíl mezi klasickými učícími se algoritmy umělé inteligence a hlubokým učením spočívá mimo jiné v předzpracování vstupních dat. Klasické algoritmy zpracovávají předem extrahované hodnoty z obrazu, zvuku, textu. U hlubokého učení je vstupem algoritmu celý obraz, zvuk nebo text a jejich zpracování probíhá samotným algoritmem hlubokého učení bez předchozího zpracování.

1.3.1 Možnosti použití algoritmů hlubokého učení

Pod pojmem hluboké učení se skrývá řada algoritmů, mezi nimiž jsou nejznámnější:

- konvoluční neuronové sítě (CNN - Convolutional Neural Networks),
- omezené Boltzmannovy systémy (RBM – Restricted Boltzmann Machines),
- tzv. „deep belief“ sítě (DBN – Deep Belief Networks),
- rekurentní neuronové sítě (RNN - Recurrent Neural Network),
- auto-enkodéry (SAE – Stacked Auto-encoder).

Do současné doby bylo hlubokého učení použito při řešení následujících problémů: Rozpoznávání různých druhů jídla z obrazů bylo realizováno s pomocí konvolučních neuronových sítí v [69], kde byla síť trénována s využitím obrazů z databáze ImageNet [70], která dosahovala 78 % přesnosti detekce, při rychlosti zpracování 30 ms na snímek. Metody pro optimalizaci výpočetní architektury algoritmů hlubokého učení byly popsány v [71]. Rozpoznání výrazu obličeje v reálném čase na mobilním telefonu bylo řešeno v [72], kde byla použita CNN, která byla trénována na GPU. V [73] je popsána implementace prostředí DjiNN pro superpočítače skládajících se z GPU serverů. Toto prostředí DjiNN slouží pro zpracování obrazu, řeči i jazyka.

Hluboké učení se začalo využívat i pro zpracování nestrukturovaného textu nebo audia. V práci [74] byl pro klasifikaci textů z čínských webových stránek použit auto-enkodér, který výrazně snížil výpočetní dimensionalitu. Práce [75] se zabývala zpracováním biomedicínských textů s použitím hlubokého učení. Konvoluční sítě byly použity pro zpracování a klasifikace audio signálů z různých prostředí v [76] nebo v [77] byl vytvořen systém pro klasifikaci různých zvukových událostí.

Další část textu bude zaměřena na popis konvoluční neuronové sítě, protože tento algoritmus byl použit při řešení v další části práce.

1.3.2 Konvoluční neuronové sítě

Pro klasifikaci obrazů do definovaných kategorií se v dnešní době nejvíce využívají konvoluční neuronové sítě, které jsou vhodné pro svou výpočetní a paměťovou nenáročnost (v porovnání s klasickými neuronovými sítěmi) i celkovou klasifikační přesnost [78]. Konvoluční sítě byly podrobně představeny v roce 1989 [79], kdy byly využity pro klasifikaci ručně psaných číslic. Vlastností těchto sítí je, že zpracovávají na svém vstupu 2D obrazová data a v určité své části používají operaci konvoluce místo klasického násobení, jako tomu je u neuronových sítí. Dále mají oproti klasickým neuronovým sítím další podstatné výhody: jednou z nich je řídká interakce mezi neurony jednotlivých vrstev, kde díky použití konvolučních jader, jsou detekovány hrany nebo jiné příznaky a tím sníženo množství operací a parametrů, než by tomu je u klasické neuronové sítě. Dalšími výhodami je sdílení některých parametrů nebo ekvivalentní reprezentace [80].

Konvoluční neuronová síť se skládá z vrstvy vstupní, výstupní, jedné nebo více vrstev konvolučních, sdružovacích vrstev (pooling layers, sub–sampling layers) a plně propojené vrstvy. Pro nastavování vah neuronové sítě slouží algoritmus zpětné propagace (back propagation) [79].

Konvoluční vrstva

Konvoluční vrstva je základním blokem konvoluční sítě. V rámci celé sítě se těchto vrstev může nacházet i více. Každá vrstva má definován počet konvolučních filtrů, které jsou optimalizovány během trénování algoritmem zpětné propagace. Výstupem vrstev jsou příznaky, které dohromady dávají mapu příznaků (feature map). Výstupní mapa příznaků r_j^ℓ je definována podle vzorce [81]:

$$r_j^\ell = f\left(\sum_{i \in M_j^\ell} \left(r_i^{\ell-1} * k_{ij}^\ell + b_j^\ell\right)\right), \tag{1.13}$$

kde j označuje pořadí mapy pro konvoluční vrstvu ℓ. f označuje nelineární aktivační funkci, b_j^ℓ označuje odchylku (bias), k_{ij}^ℓ je konvoluční filtr použitý v předcházející vrstvě a M_j^ℓ označuje množinu všech map pro ℓ-tou vrstvu.

Sdružovací vrstva

Sdružovací vrstva (pooling layer, sub–sampling layer) se používá pro snížení počtu prvků (tzv. pod vzorkování) výstupu předcházející vrstvy. Touto vrstvou bývá zpravidla vrstva konvoluční. Při redukování počtu prvků se používá maximální nebo průměrovací funkce, kde po aplikaci zvolené funkce na část prvků (hodnot pixelů), je výstupem jedna hodnota.

Plně propojená vrstva

Slouží k propojení všech výstupů předchozí vrstvy do jednoho výstupu, tak jak tomu je u běžných neuronových sítí. Výstup této plně propojené vrstvy je popsán rovnicí [81]:

$$r^{\ell} = f\left(\mathbf{W}^{\ell}\mathbf{r}^{\ell-1} + \mathbf{b}^{\ell}\right), \tag{1.14}$$

kde f označuje výstupní aktivační funkci, ℓ v tomto případě označuje poslední vrstvu v síti, $\mathbf{W}^{\ell}$ označuje matici vah, výstup předcházející vrstvy je označen $\mathbf{r}^{\ell-1}$ a $\mathbf{b}^{\ell}$ označuje vektor odchylky (bias).

Algoritmus zpětné propagace

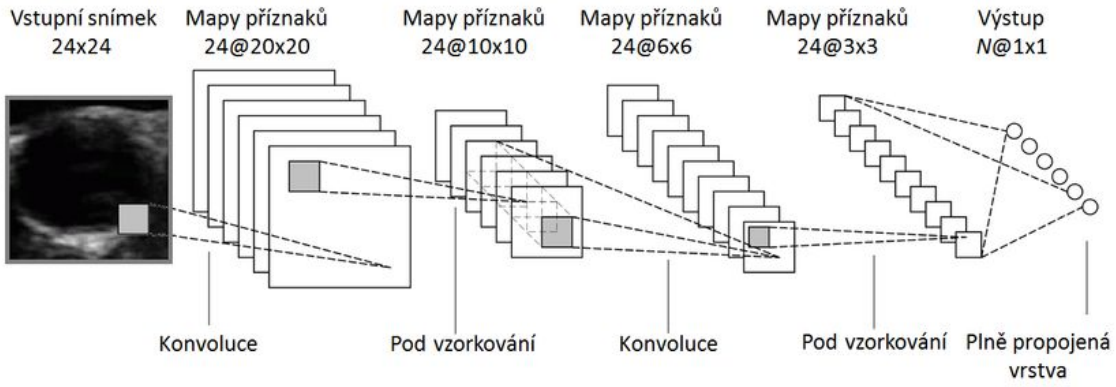
Algoritmus zpětné propagace chyb [79] (back–propagation) je běžným algoritmem pro trénování neuronových sítí a je založen na výpočtu gradientu ztrátové funkce pro všechny váhy neuronové sítě. Ztrátová funkce je spočtena jako rozdíl mezi aktuálním a očekávaným výstupem pro všechny trénovací prvky. Ztrátová funkce, založená na výpočtu střední kvadratické chyby E pro C kategorií a N trénovacích prvků, je popsána rovnicí [81]:

$$E = \frac{1}{2}\sum_{n=1}^{N}\sum_{c=1}^{C}\left(t_c^n - g_c^n\right)^2, \tag{1.15}$$

kde t_c^n označuje c-tý prvek z množiny všech kategorií (label) pro n-tý trénovací prvek. g_c^n je spočteným výstupem sítě. Při učení je cílem tuto chybu minimalizovat, proto se na základě vypočtené chyby mění jednotlivé váhy neuronové sítě.

Architektura konvoluční neuronové sítě

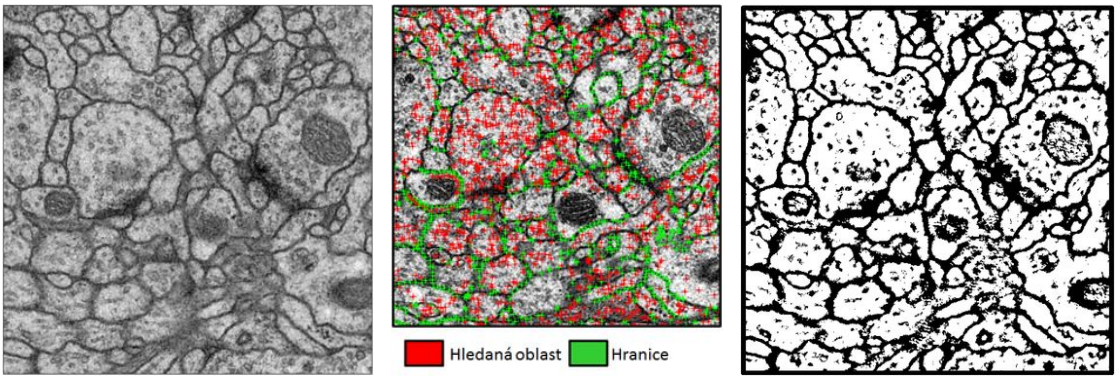
Jedno z možných sestavení konvoluční neuronové sítě je zobrazeno na obrázku 1.7. Tato síť se skládá ze dvou konvolučních a dvou sdružovacích vrstev. Následuje vrstva plně propojená a výstupní, která určuje pravděpodobnosti pro N klasifikačních tříd, na kterých byla daná síť trénována. Konvoluční vrstva se skládá z 24 různých filtrů o velikostech 5x5 pixelů, které jsou aplikovány na vstupní obraz o velikosti 24x24 pixelů, dále jsou tyto konvolvované výstupy pod vzorkovány na velikost 10x10 pixelů. Po provedení následných opakujících se operací konvoluce a pod vzorkování jsou výstupní mapy příznaků sloučeny v plně propojené vrstvě. Výstupem sítě jsou pravděpodobnostní hodnoty každé kategorie.



Obr. 1.7: Schéma konvoluční neuronové sítě.

1.4 Trénovatelná segmentace

Trénovatelná segmentace [82] kombinuje učící se algoritmy umělé inteligence spolu s vybranými obrazovými příznaky za účelem vytvořit segmentovaný obraz, kde jsou jednotlivé části segmentů od sebe odlišeny buď binárně, nebo v odstínech šedé. Během trénovací části procesu segmentace, jsou extrahovány obrazové příznaky popisující vybrané pixely a jejich nejbližší okolí, kde každý vybraný pixel byl předem expertem označen buď jako positivní nebo jako negativní v závislosti na řešeném problému.



(a) Originální snímek (b) Ohodnocený snímek (c) Segmentovaný výstup

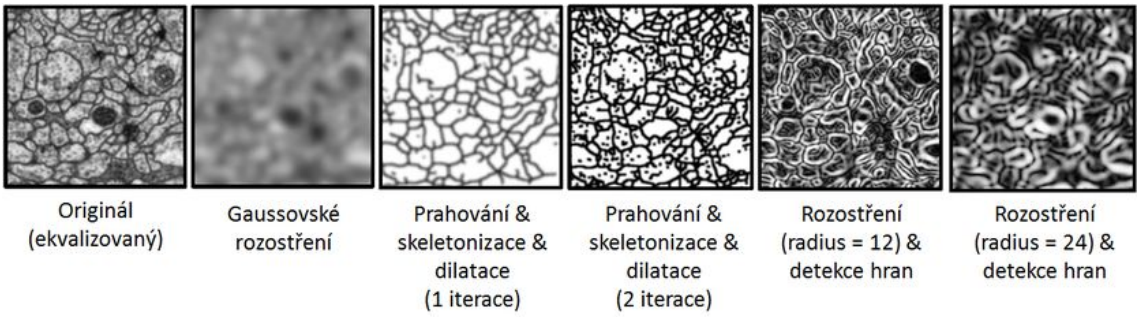
Obr. 1.8: Zpracování snímku trénovatelnou segmentací.

Alternativou k metodě trénovatelné segmentace jsou interaktivní segmentační metody, které již byly v minulosti použity v mnoha pracích. Příkladem může být interaktivní segmentační metoda založená na principu oříznutí grafu (graph-cut). Na základě této populární metody vzniklo mnoho odvozených variant [83, 84]. V [85]

bylo vytvořeno segmentační prostředí kombinující metodu oříznutí grafu s metodou náhodné cesty (random walks) [86]. V [87] byla použita jiná metoda založená na zjišťování podobností v obrazech, za účelem segmentace scén z video sekvencí.

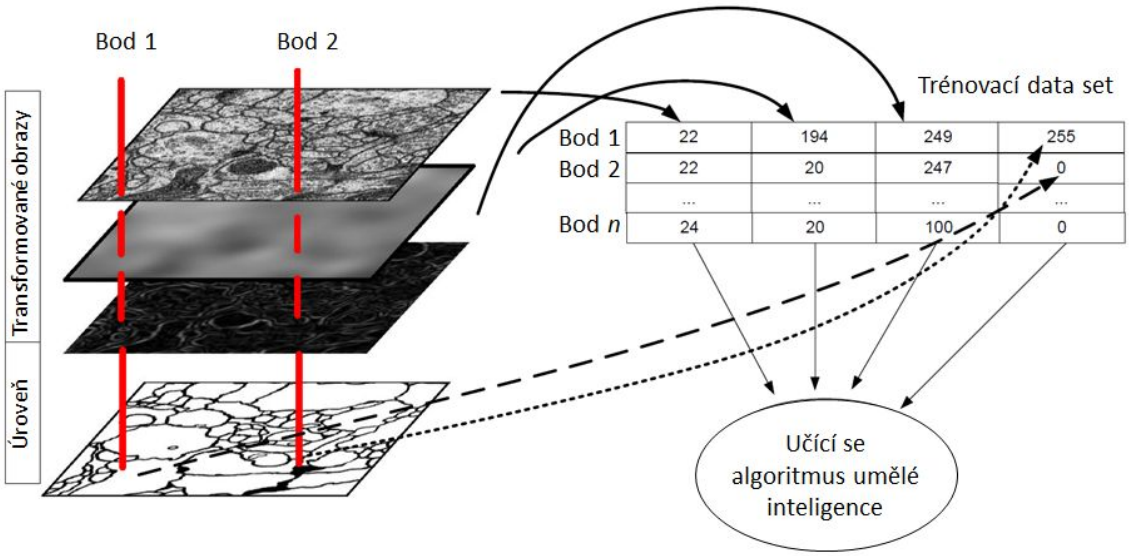
1.4.1 Princip trénovatelné segmentace

Pro názorný příklad může být uvedena segmentace obrazů mozku mouchy z elektronového mikroskopu [6], kde bylo cílem oddělit jednotlivé buňky mozku od sebe na základě jejich společných hranic. Nejprve jsou na vstupním obrázku označeny části buněk (positivní oblasti) a dále části hranic mezi buňkami (negativní oblasti) viz obrázek 1.8b. Tímto jsou definovány trénovací oblasti. Po té následuje předzpracování vstupního obrazu několika vybranými obrazovými transformacemi, díky kterým jsou získány o pixelu a jeho okolí nové informace. Příkladem transformací může být Gaussovské rozostření, detekce hran, skeletonizace, maximální, minimální hodnoty v rámci částí obrazu, škálovatelné prostorové transformace (scale–space). Příklady vybraných transformací aplikovaných na vstupní obraz jsou zobrazeny na obrázku 1.9.



Obr. 1.9: Ukázka transformace obrazů.

Kombinací uživatelem označených trénovacích oblastí s obrazovými příznaky vznikne vstupní soubor dat, který je použit učícími se algoritmy umělé inteligence (např. k-NN, SVM, rozhodovací stromy, náhodné lesy,...). Celý princip trénování je zobrazen na obrázku 1.10, kde jsou v místech uživatelem zvolených bodů vyčteny hodnoty pro jednotlivé transformace a dále je vše převedeno do tabulkového formátu, který je vstupem učícího se algoritmu. Zvolený učící se algoritmus poté natrénuje požadovanou znalost na základě daného problému a v testovací fázi klasifikuje nový neznámý snímek (viz obrázek 1.8c), kde od sebe oddělí jednotlivé buňky jejich hranicemi. V tomto konkrétním příkladě jsou ještě pro zlepšení výsledků použity další obrazové metody. Takto je natrénovaná znalost (model) aplikována soubor všechny 2D obrazy, které jako celek tvoří 3D obraz.



Obr. 1.10: Princip trénovací fáze trénovatelné segmentace.

Algoritmus trénovatelné segmentace byl již použit autorem této práce pro řešení problému detekce domů ze satelitních snímků v [88] nebo v [9] byla řešena segmentace obrazových dat meteoritů. Dále byla vytvořena a popsána platforma pro trénovatelnou segmentaci [8] a v [6] byla řešena segmentace medicínských snímků mozku mouchy, které byly vytvořeny elektronovým mikroskopem, jak již bylo popsáno výše.

2 CÍLE DISERTACE

Cílem disertační práce bylo navrhnout původní algoritmy strojového učení pro získávání znalostí z multimediálních dat, které budou na vytipovaných příkladech dosahovat přesnější výsledky, než současné přístupy. Některé algoritmy budou navrženy pro spuštění v distribuovaných systémech, čímž dojde k razantnímu urychlení celého procesu a ušetření značného množství elektrické energie. Výsledkem bude, že díky tomuto urychlení bude možné použít větší množství vstupních dat nebo spustit algoritmus vícekrát s různými parametry za účelem natrénovat přesnější modely pro získávání znalostí z multimediálních dat.

Pojmem multimediální data jsou zde myšlena obrazová data, textová data, audio data a video sekvence. Navržené algoritmy budou zpracovávat přímo tato nestrukturovaná data a dále data, která byla z těchto nestrukturovaných dat získána.

Hlavními cíli disertace tedy bude:

- Vytvoření a optimalizace vybraných současných metod za účelem jejich akcelerace. Tyto vytvořené metody budou určeny pro nasazení na speciálních paralelních zařízeních, která se vyznačují vysokým výpočetním výkonem a nízkou spotřebou ve srovnání s běžnými výpočetními procesorovými stanicemi.
- Vytvořit komplexní systém pro trénování a testování objektového detektoru, který díky vytvořeným optimalizovaným algoritmům bude fungovat dostatečně rychle a díky vytvořeným variantám několika obrazových příznaků (Haar, HOG, LBP) bude určen pro řešení různých typů problémů.
- Rozšířit systém pro detekci objektů v obrazech o algoritmy hlubokého učení, které budou sloužit pro kategorizaci již detekovaných částí snímků.
- S použitím evolučních algoritmů provést optimalizaci výběru parametrů kaskádového modelu pro objektový detektor za účelem zvýšení přesnosti.
- Všechny algoritmy implementovat do jednoho systému, kde je bude možné vzájemně a jednoduše kombinovat a jejich použití bude automatizované.
- Distribuování vybraných výpočetních algoritmů mezi skupinu výpočetních stanic za účelem zrychlení celkového výpočetního času.
- Rychlostní optimalizace učícího se algoritmu k-nejbližších sousedů pro zpracování velkých objemů dat.
- Pro algoritmus trénovatelné segmentace optimalizovat testovací proces pro zpracování většího množství snímků o vyšším rozlišení. Tímto algoritmem poté řešit složitější problémy obrazové segmentace.
- Připravit nové databáze a ověřit funkčnost algoritmů na dalších vybraných problémech. Otestovat rychlost vytvořených algoritmů pro grafické akcelerátory na velkých databázích.
- Na vybraných příkladech dosáhnout lepších výsledků, než jsou dosaženy za

použití metod známých v současnosti.

- Provést srovnání dosažených výsledků s dosavadními přístupy.

V disertační práci budou popsána navržená řešení. Dále budou provedeny testy navržených metod na vybraných příkladech.

3 NAVRŽENÉ METODY ŘEŠENÍ

Navržená řešení se zabývají výběrem a optimalizací metod pro získávání znalostí z multimediálních dat. Jedná se o optimalizaci algoritmů vytvořených v rámci systému pro detekci a klasifikaci objektů v obrazech, dále o optimalizované verze učících se algoritmů umělé inteligence vhodných pro zjišťování znalostí z báze dat a nakonec o optimalizování testovací části algoritmu trénovatelné segmentace.

Schémata zobrazená v rámci této kapitoly obsahují bloky, které označují určitou vlastnost části metody. Vstupy jsou označovány modrou barvou a výstupy červenou barvou. Bloky, které jsou ohraničeny hnědou barvou, označují vytvořenou optimalizovanou verzi algoritmu pro paralelní hardware. Zeleně ohraničené bloky označují, o jaké části byly současné metody rozšířeny.

3.1 Systém pro detekci objektů v obrazech

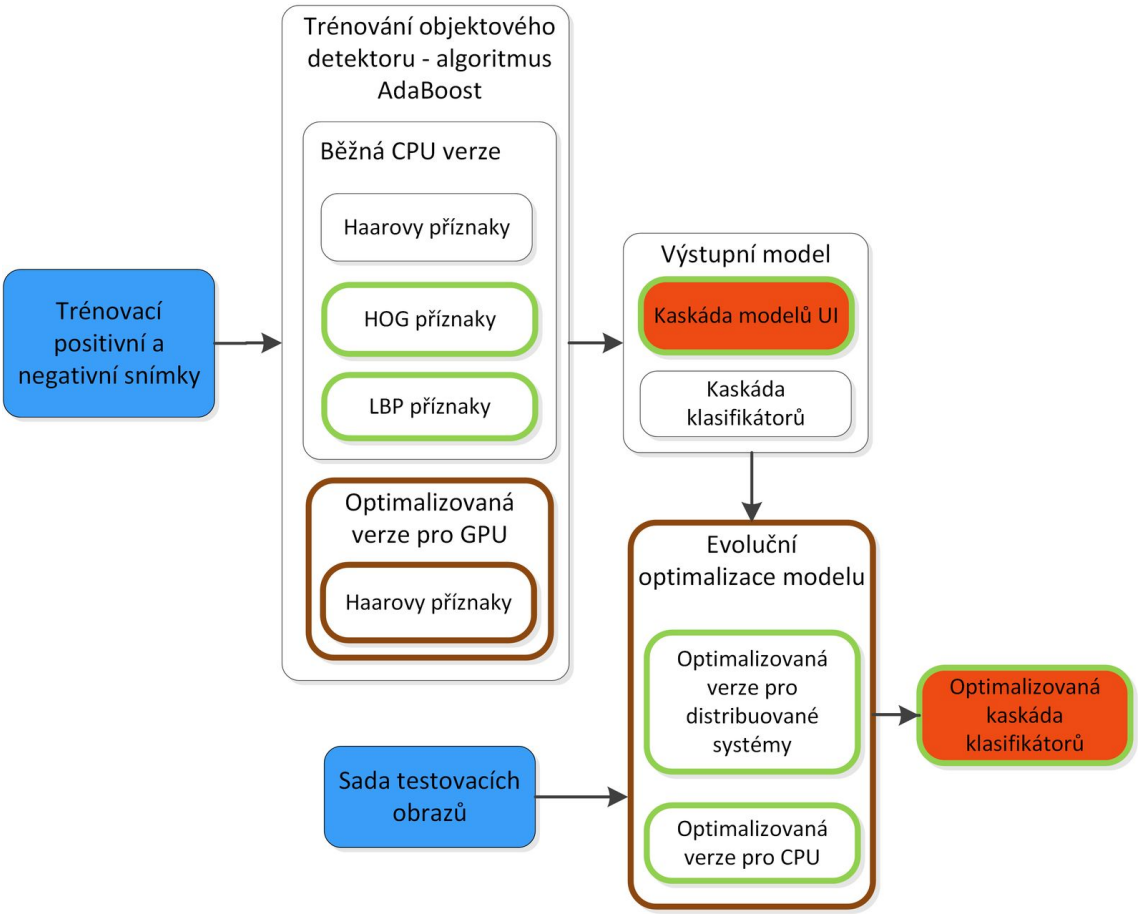
V rámci práce byl vytvořen komplexní systém pro trénování a testování objektového detektoru, který díky algoritmům speciálně navrženým pro distribuované systémy spolu s kombinací několika variant obrazových příznaků (Haar, HOG, LBP) a evoluční optimalizací při trénování kaskádního modelu a který dokáže dosáhnout lepších výsledků při detekci objektů.

Tento systém se skládá ze tří hlavních částí, kde první část slouží k natrénování kaskádního modelu pro detekci objektů (viz obrázek 3.1), druhá část je určena k trénováním modelu pro klasifikaci různých typů detekovaných obrazů s pomocí hlubokého učení (viz obrázek 3.2) a poslední část (viz obrázek 3.3) slouží k detekci objektů na základě natrénovaného kaskádního modelu a ke klasifikaci takto detekovaných objektů do kategorií.

Trénování detektoru

Schéma trénování kaskádního modelu je zobrazeno na obrázku 3.1. Vstupem pro trénování kaskádního modelu jsou trénovací positivní a negativní snímky, které jsou předzpracovány a s pomocí algoritmu AdaBoost jsou na základě zvolených obrazových příznaků (Haar, HOG, LBP) vybírány ty s nejmenší trénovací chybou. V případě Haarových příznaků byla vytvořena verze algoritmu schopná běžet na paralelním hardware (GPU akcelerátorech), kde pro případ velkých trénovacích data setů bylo dosaženo až 150-ti násobného zrychlení. Pro zbylé dvě verze příznaků byl algoritmus upraven, aby mohl efektivně běžet na více jádrových procesorech. Nejlepší vybrané příznaky jsou dále zpracovány podle typu zvoleného výstupního modelu. V případě, kdy je zvolen jako výstupní model kaskáda modelů umělé inteligence,

jsou hodnoty extrahovaných příznaků použity pro trénování jednotlivých kaskádních modelů umělou inteligencí (k–NN, SVM, rozhodovací strom, ...). Evoluční optimalizace pro kaskádu modelů nebyla vytvořena z důvodu vysoké časové výpočetní náročnosti. Pokud je vybrána pro výstupní model kaskáda klasifikátorů, jsou nejlepší příznaky seskupeny do kaskády slabých lineárních klasifikátorů a následuje dodatečné nastavení parametrů této kaskády pomocí evolučních algoritmů. Vstupem těchto evolučních algoritmů je sada testovacích obrazů a výstupem je optimalizovaná kaskáda klasifikátorů. Celá problematika bude více popsána v kapitole 3.1.2.

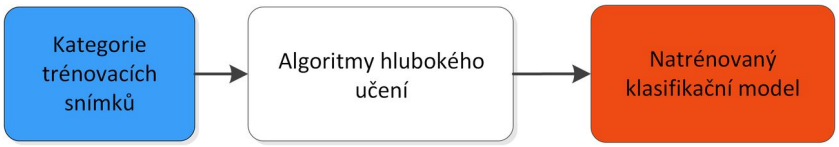


Obr. 3.1: Schéma systému pro trénování detektoru.

Trénování klasifikačního modelu

Protože celý systém není zaměřen pouze na detekci objektů, ale i na klasifikaci takto detekovaných objektů, byl za pomoci algoritmů hlubokého učení vytvořen proces, který dokáže od sebe odlišit různé objekty detekované jedním detektorem, který detekuje zejména na základě tvaru, případně textury. Příkladem může být detekce příčného řezu arterie z ultrazvukového snímku, kde bude kromě lokalizace arterie

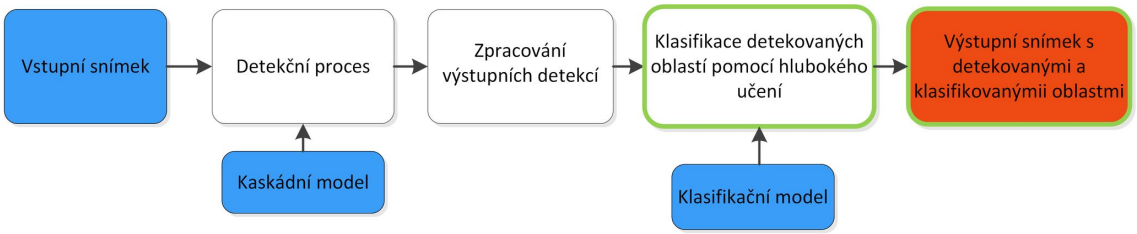
ve snímku určeno například, jestli se jedná o zdravou artérii nebo jestli obsahuje například určité množství plaku, který může predikovat různá onemocnění. Schéma procesu je zobrazeno na obrázku 3.2. Vstupem jsou kategorie trénovacích snímků: například zdravé arterie, arterie s nízkým množstvím plaku a arterie s vyšším množstvím plaku. Dále je pomocí algoritmů hlubokého (v této práci byly použity konvoluční neuronové sítě) učení natrénován klasifikační model, který je později použit v detekčním procesu. Problematika trénování pomocí hlubokého učení bude podrobněji popsána v kapitole 3.1.5.



Obr. 3.2: Schéma pro trénování modelu pro klasifikaci detekovaného objektu metodami hlubokého učení.

Detekční a klasifikační proces

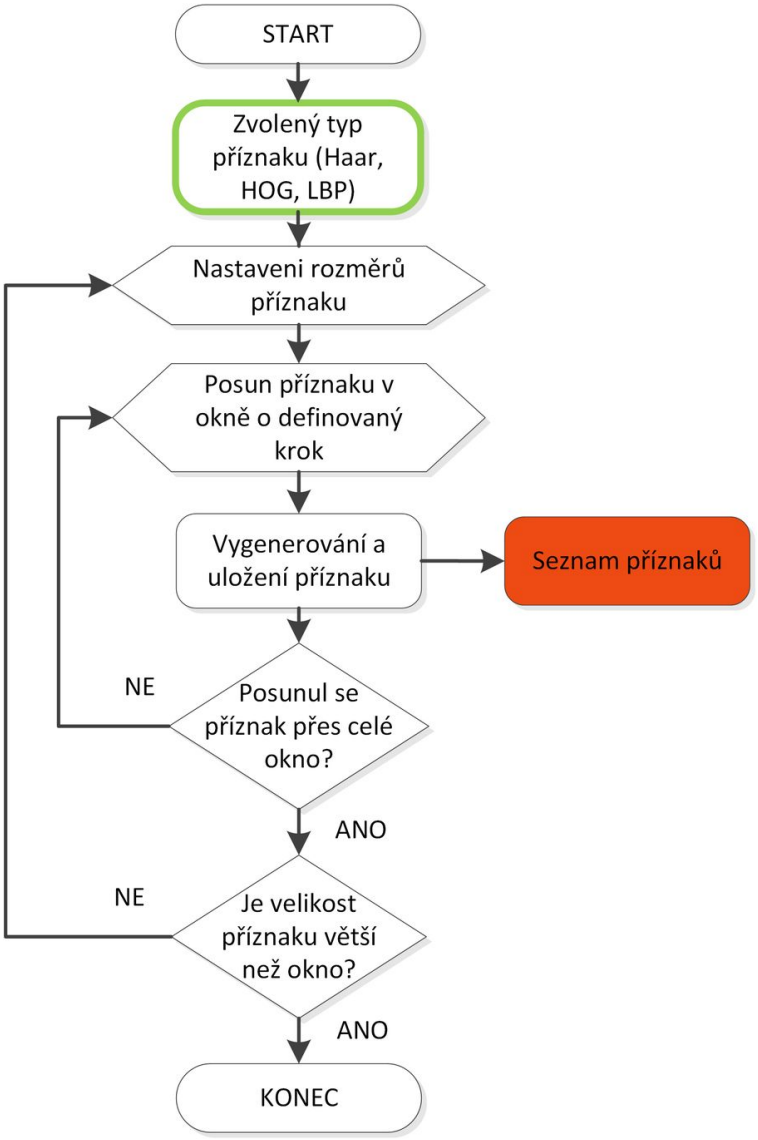
Pokud je již natrénován potřebný kaskádní a klasifikační model, kde jejich trénování bylo časově a paměťové náročné, následuje aplikování těchto modelů, které by mělo být na druhou stranu provedeno nejlépe v reálném čase. Schéma procesu je zobrazeno na obrázku 3.3. Vstupem je snímek, v kterém budou hledány a klasifikovány objekty. Dalšími vstupy jsou natrénované modely pro klasifikaci a predikci objektů. Po načtení snímků následuje jeho předzpracování, poté jsou na základě natrénovaného kaskádního modelu detekovány objekty. Protože každý objekt bývá zpravidla detekován mnohonásobně, musí být výstupní detekce dále zpracovány a pro každý objekt vytvořena jedna, ta nejvíce vypovídající. Takto detekované objekty jsou dále klasifikovány klasifikačním modelem a výstupem je obraz s detekovanými objekty a určenou kategorií každého objektu. Detekční proces bude dále popsán v kapitole 3.1.6.



Obr. 3.3: Schéma detekčního a klasifikačního procesu.

3.1.1 Vytvořené obrazové příznaky

Pro trénování kaskádního modelu byly vytvořeny obrazové příznaky založené na metodách Haar, HOG a LBP. Tyto příznaky byly rozšířeny oproti běžným variantám o další parametry, díky kterým bylo těchto příznaků vygenerováno více. Při aplikování každého typu příznaků musí být vždy vstupní obraz transformován. V případě použití Haarových a LBP příznaků se jedná o transformaci na integrální obraz. Z takto upraveného obrazu pak daný příznak počítá odezvu, což může být buď jedna hodnota, nebo histogram hodnot v závislosti na použitém typu příznaků.



Obr. 3.4: Obecné schéma generování příznaků.

Princip generování příznaků je zobrazen schématicky na obrázku 3.4. Příznaky jsou generovány v rámci rozměrů trénovacího okna, jehož rozměry musí být na začátku specifikovány. Ve vybrané řešeném případu byl zvolen rozměr 24x24 pixelů.

Při samotném generování příznaků je nejprve nastaven nejmenší možný počáteční rozměr příznaku, který je poté iterativně zvětšován a posouván ve vertikálním a horizontálním směru v rámci rozměrů trénovacího okna. Každý takto vygenerovaný příznak je uložen do seznamu pro další použití při trénování.

Haarovy příznaky

V rámci práce byla vytvořena implementace podle klasické metody Viola–Jones, kde byly použity hranové, čárové a středové příznaky a pro velikost detekčního okna 24x24 pixelů jich bylo vygenerováno 189 664. V rámci následného testování jednotlivých příznaků na vybraných příkladech bylo zjištěno, že Haarovy příznaky dosahují nižší výpočetní náročnosti a lepší přesnosti než zbylé dva typy příznaků.

Příznaky histogramů orientovaných gradientů

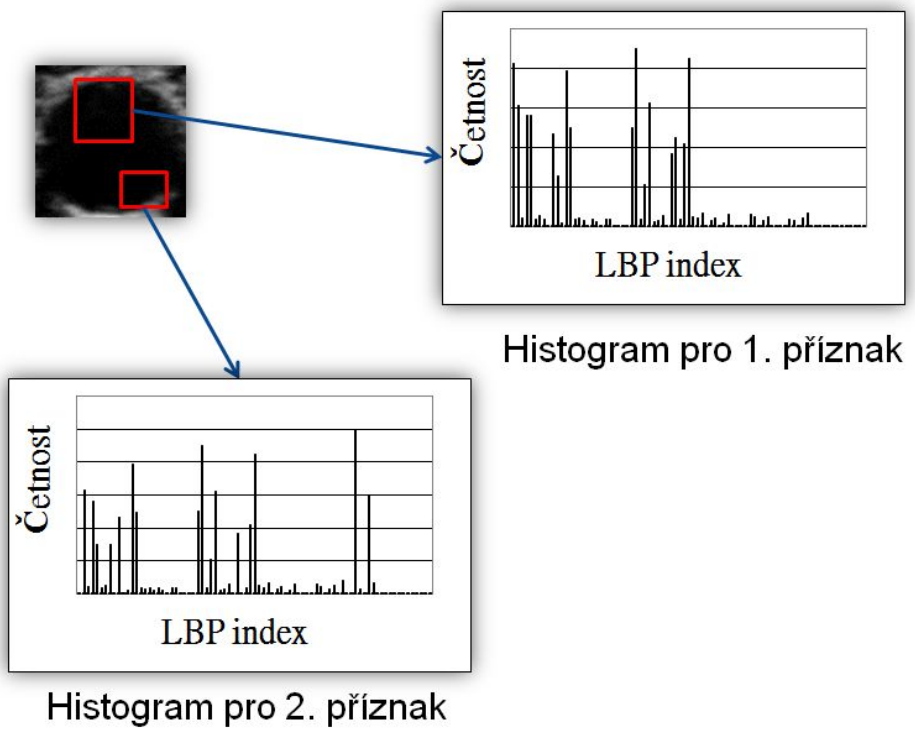
Na základě metody histogramů orientovaných gradientů byly vytvořeny obrazové příznaky. Oproti běžnému přístupu byly tyto příznaky rozšířeny o možnost být generovány pro různé rozměry v rámci trénovacího okna. Příznaky jsou aplikovány na vstupní obraz, který musí být předzpracován: nejprve je provedena detekce hran, dále jsou spočteny velikosti a směry jednotlivých gradientů. Odezvou příznaku na obraz je histogram gradientů. Celkový počet vygenerovaných HOG příznaků je 10 000.

Lokální binární příznaky

Lokální binární příznaky byly vytvořeny na základě metody lokálních binárních vzorů (LBP). Oproti původnímu přístupu jsou tyto příznaky škálovány přes celé trénovací okno. Dalším vylepšením bylo použití integrálního obrazu pro urychlení výpočtu výstupního histogramu vzorů. Pro každý pixel v definované části obrazu (určeno LBP příznakem) je spočítán histogram výskytů LBP vzorů (viz obrázek 3.5). Tato hodnota je tedy výstupní odezvou LBP příznaku na obraz. Pro velikost detekčního okna 24x24 pixelů je vygenerováno 1458 LPB příznaků.

3.1.2 Trénování detektoru

Tato kapitola popisuje princip trénování detektoru (kaskádního modelu) na základě zvolených vstupních příznaků a typu výstupního modelu. Vytvořený systém vychází z původního přístupu Violy a Jonese. V této práci byl ale rozšířen o možnosti použití dalších příznaků (HOG, LBP) nebo dále o možnost trénovat kaskádu modelů algoritmy umělé inteligence (pro každý stupeň kaskády je natrénován model na vybraných nejlepších příznacích s použitím učících se algoritmů - např. k-NN,



Obr. 3.5: Výsledný histogram pro určité příznaky.

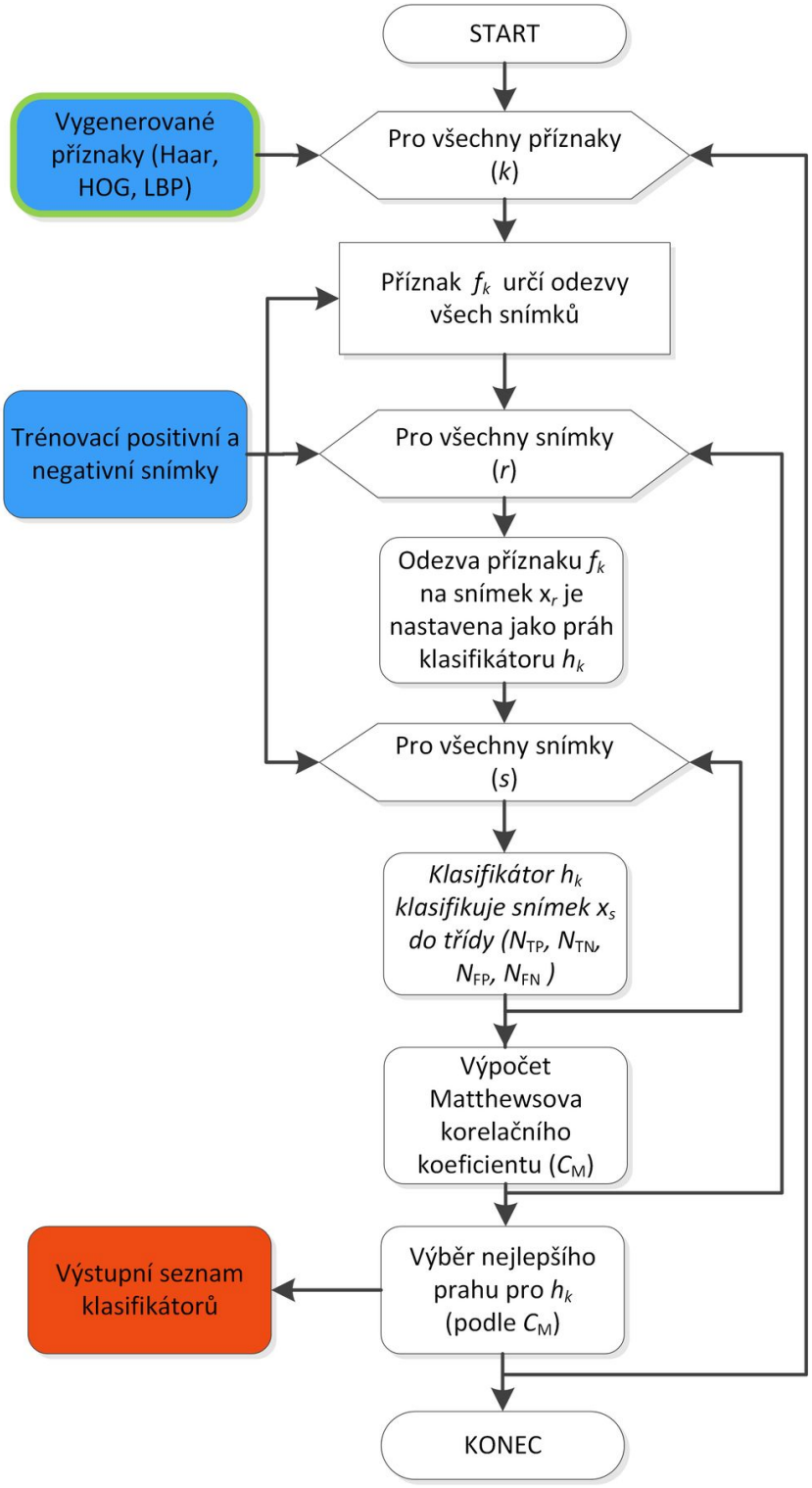
SVN, ...). Některé části systému pro trénování byly optimalizovány pro spuštění na speciálním paralelním hardware a tím bylo dosaženo výrazného urychlení výpočtů.

Samotný trénovací proces začíná načtením vstupní sady positivních (snímky arterie) a negativních (snímky pozadí) snímků, které jsou dále podle typu vybraných příznaků předzpracovány. Dalším krokem je vygenerování všech příznaků, z kterých po spočtení definovaných parametrů, vzniknou slabé klasifikátory.

Výpočet parametrů klasifikátorů

Cílem tohoto procesu je k vygenerovaným příznakům spočítat parametry. Tím se z příznaků stanou slabé klasifikátory. Princip výpočtu je zobrazen na obrázku 3.6. Vstupem procesu jsou vygenerované příznaky a trénovací positivní a negativní snímky. Výstupem je seznam lineárních klasifikátorů. V průběhu tohoto procesu je cílem nalézt nejvíce vypovídající prahovou hodnotu a polaritu pro každý klasifikátor. Pro výpočet nejlepší prahové hodnoty klasifikátoru byl použit Matthewsův korelační koeficient C_{M}, který je spočten na základě zařazení trénovacích snímků do tříd podle rovnice:

$$C_{\mathrm{M}} = \frac{N_{\mathrm{TP}}\, N_{\mathrm{TN}} - N_{\mathrm{FP}}\, N_{\mathrm{FN}}}{\sqrt{(N_{\mathrm{TP}} + N_{\mathrm{FP}})\,(N_{\mathrm{TP}} + N_{\mathrm{FN}})\,(N_{\mathrm{TN}} + N_{\mathrm{FP}})\,(N_{\mathrm{TN}} + N_{\mathrm{FN}})}}, \tag{3.1}$$



Obr. 3.6: Schéma výpočtu parametrů klasifikátorů.

kde N_{TP} označuje správně klasifikované positivní snímky, N_{TN} označuje správně klasifikované negativní snímky, N_{FP} označuje falešně positivní snímky a N_{FN} označuje falešně negativní snímky. Dále je nalezena prahová hodnota a polarita klasifikátoru s největší absolutní hodnotou C_{M} a ta je přiřazena aktuálně zpracovávanému klasifikátoru. Podle znaménka C_{M} je určena polarita aktuálního klasifikátoru.

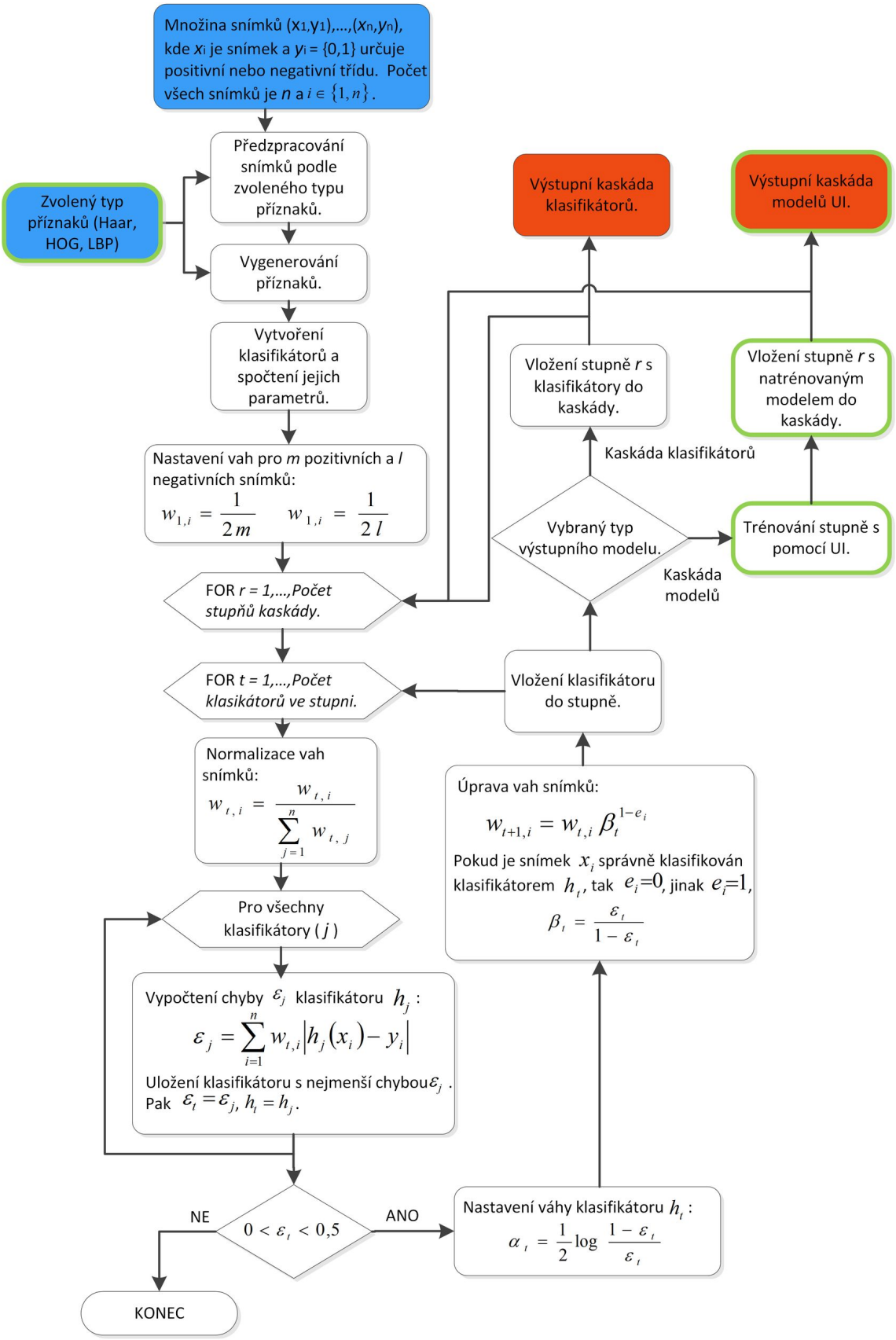
V případě zvolení Haarových příznaků je tímto způsobem vytvořeno 189 664 klasifikátorů. Při použití trénovacích dat v počtu desítek tisíc je tento proces velmi výpočetně náročný, proto byla funkce pro výpočet parametrů klasifikátorů optimalizována pro nasazení na speciálních grafických akcelerátorech, kde bylo dosaženo až 150-ti násobného zrychlení. Tato optimalizace bude podrobněji popsána v kapitole 3.1.3.

Proces trénování

Celý trénovací proces je zobrazen na obrázku 3.7. Tento postup byl vytvořen pro paralelní běh na více procesorech. Poté co jsou vygenerovány všechny příznaky, začíná vyhledávání nejlepších lineárních klasifikátorů pomocí algoritmu AdaBoost [89, 90]. Tento algoritmus nejprve nastaví startovní váhy pro trénovací snímky. Dále již probíhá výběr definovaného počtu nejlepších klasifikátorů. Nejprve jsou váhy trénovacích snímků normalizovány, poté je vybrán klasifikátor, který dosáhl při klasifikaci trénovacích snímků nejnižší chyby ϵ_t. Pokud je chyba $\epsilon_t \geq 0,5$ je trénovací proces ukončen, jinak jsou vybranému klasifikátoru nastaveny váhy α_t a zvětšeny váhy špatně klasifikovaným trénovacím snímkům. Vybraný klasifikátor je vložen do stupně. Takto je vybranými klasifikátory naplněn každý stupeň kaskádního modelu. V případě, že je jako výstupní model použita kaskáda modelů umělé inteligence, je ještě navíc v rámci každého stupně provedeno trénování modelu umělé inteligence na základě vstupních příznaků, které jsou součástí lineárních klasifikátorů. Výstupem trénovacího procesu je tedy podle vybraného typu: kaskáda klasifikátorů nebo kaskáda modelů umělé inteligence.

Kaskádní model

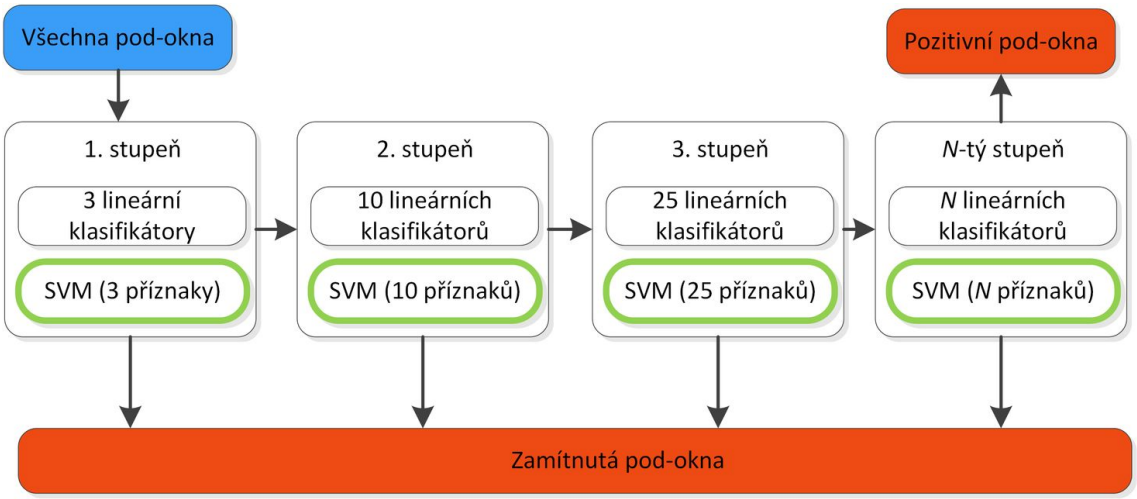
Výstupem trénovacího procesu je kaskádní model. Jeho účelem je výrazně urychlit detekční proces tím, že na každém stupni je zamítnut určitý počet negativních pod oken a propuštěna positivní pod okna. V této práci byla původní verze [91] kaskádního modelu (kaskády klasifikátorů) rozšířena o možnosti použití modelů umělé inteligenci místo lineárních klasifikátorů v rámci každého stupně kaskádního modelu. Princip funkce kaskády pro klasické lineární klasifikátory i pro modely umělé inteligence je vysvětlen na obrázku 3.8, kde je pro každý stupeň použit definovaný



Obr. 3.7: Schéma procesu trénování kaskádního modelu.

počet lineárních klasifikátorů nebo je v druhém případě trénován SVM model na definovaném počtu příznaků.

V rámci trénovacího procesu jsou tedy výstupem dva typy kaskádního modelu: kaskáda klasifikátorů (je rychlejší, ale zase jednodušší) a dále kaskáda modelů umělé inteligence (je relativně pomalá). Při trénování by kaskádní model měl být natrénován tak, aby dosahoval vysoké úspěšnosti detekce a zároveň vykazoval nízký počet falešně positivních detekcí.



Obr. 3.8: Schéma kaskádního zapojení.

3.1.3 Optimalizace trénovacího procesu

V rámci práce byly výpočetně nejnáročnější části trénovacího procesu optimalizovány pro spuštění na grafických akcelerátorech. Grafické akcelerátory na rozdíl od běžných procesorů obsahují až tisíc krát více výpočetních jader, která ale na druhou stranu zvládají pouze jednoduší instrukce a jsou i pomalejší. V případě použití velkého množství trénovacích snímků (řádu desítek tisíc) je možné díky optimalizovanému trénovacímu procesu urychlit výpočet z několika týdnů na několik hodin. Tento optimalizovaný proces může být distribuován mezi více grafických akcelerátorů souběžně. Optimalizované funkce byly vytvořeny za pomocí prostředí NVIDIA CUDA [29]. Z důvodů složitějšího vývoje algoritmů pro grafické akcelerátory byla vytvořena optimalizovaná verze pouze pro Haarovy příznaky, které patří v oblasti detekce objektů mezi častěji používané. Optimalizovány byly dvě funkce: první pro určení parametrů lineárních klasifikátorů a druhá funkce optimalizovala algoritmus AdaBoost, který vybírá nejlepší klasifikátory.

Prostředí CUDA

CUDA je platforma pro paralelní výpočty na grafických akcelerátorech (GPU) od společnosti NVIDIA. V současné době existuje mnoho algoritmů vyvinutých na platformě CUDA a další se stále vyvíjejí. Syntaxe jazyka prostředí CUDA vychází z jazyka C a obsahuje navíc speciální funkce pro programování GPU.

Běžný grafický akcelerátor se skládá z (v dnešní době z tisíců) jednoduchých výpočetních jader, která jsou součástí streamovacích multi-procesorů (SMs) jejichž účelem je spouštět výpočetní vlákna paralelně. Tato vlákna jsou seskupena do bloku, který je dále součástí mřížky (tato mřížka obsahuje všechny bloky). Všechna vlákna v rámci jednoho bloku jsou spuštěna paralelně a dokáží spolu komunikovat a sdílet společnou paměť. Počty bloků a počty vláken v bloku mohou být nastaveny pro každý algoritmus rozdílně. Grafické akcelerátory jsou založeny na architektuře „jedné instrukce - mnoha vláken“ (SIMT - Single Instruction Multiple Threads) a obsahují paměť lokalizovanou na čipu (registry, sdílená paměť) a paměť lokalizovanou mimo čip (lokální, konstantní, globální, texturová). Paměti lokalizované na čipu jsou rychlejší než paměti mimo čip, ale mají omezenou velikost. Paměti lokalizované mimo čip mohou být nastaveny pouze pro čtení, což výrazně zvýší jejich rychlost.

Optimalizované funkce

Trénovací proces detekce objektů v obrazech byl optimalizován s využitím již zmíněného prostředí CUDA. Vzhledem k tomu, že celá platforma pro detekci a klasifikaci objektů v obrazech je vytvořena v programovacím jazyku JAVA, bylo třeba najít řešení pro vzájemné propojení jazyka JAVA s prostředím CUDA. K tomu byla použita knihovna JCuda[1]. Optimalizovány byly dvě nejvíce výpočetně náročné funkce: první pro hledání optimálních prahových hodnot a polarit slabých klasifikátorů (viz obrázek 3.9) a druhá (viz obrázek 3.10) pro hledání nejlepších klasifikátorů, kdy byla užita optimalizace algoritmu AdaBoost. Protože grafické akcelerátory nedokážou zpracovávat objekty z programovacího jazyka JAVA, bylo nutné všechny tyto objekty (Haarovy příznaky, integrální obrazy a klasifikátory) převést do polí typu *float*, protože grafické akcelerátory neumí v současné době efektivně pracovat s dvojitou přesností, ale pouze s jednoduchou.

První optimalizovaná funkce, která nastavuje parametry klasifikátorů, má na vstupu 189 664 Haarových příznaků a integrální obrazy všech trénovacích snímků (viz obrázek 3.9). Nejprve jsou tato data převedena na pole typu *float* a následně kopírována do paměti GPU. Po skončení výpočtu jsou klasifikátory s nastavenými parametry kopírovány zpět do CPU. Druhá optimalizovaná funkce akceleruje algoritmus AdaBoost(viz obrázek 3.10), kde je na základě vstupních integrálních obrazů

[1]Dostupné z URL: `http://jcuda.org/`

Input: Haarovy příznaky, integrální obrazy
Output: slabé klasifikátory

1 transformace integrálních obrazů
2 transformace příznaků
3 kopírování příznaků a integrálních obrazů do paměti GPU
4 **foreach** *příznak* **do**
5 | počítání prahových hodnot (na základě všech integrálních obrazů)
6 | nastavení nejlepší prahové hodnoty a polarity klasifikátoru
7 **end**
8 kopírování všech klasifikátorů do CPU

Obr. 3.9: Schéma určení prahových hodnot a polarit klasifikátorů.

a seznamu klasifikátorů vybrán v každé iteraci nejlepší klasifikátor. Tyto algoritmy byly testovány až na 4 GPU, kde nejlepší zrychlení bylo 150-ti násobné oproti klasické CPU verzi algoritmu. Podrobné testování optimalizovaného algoritmu bude popsáno v kapitole 4.4.

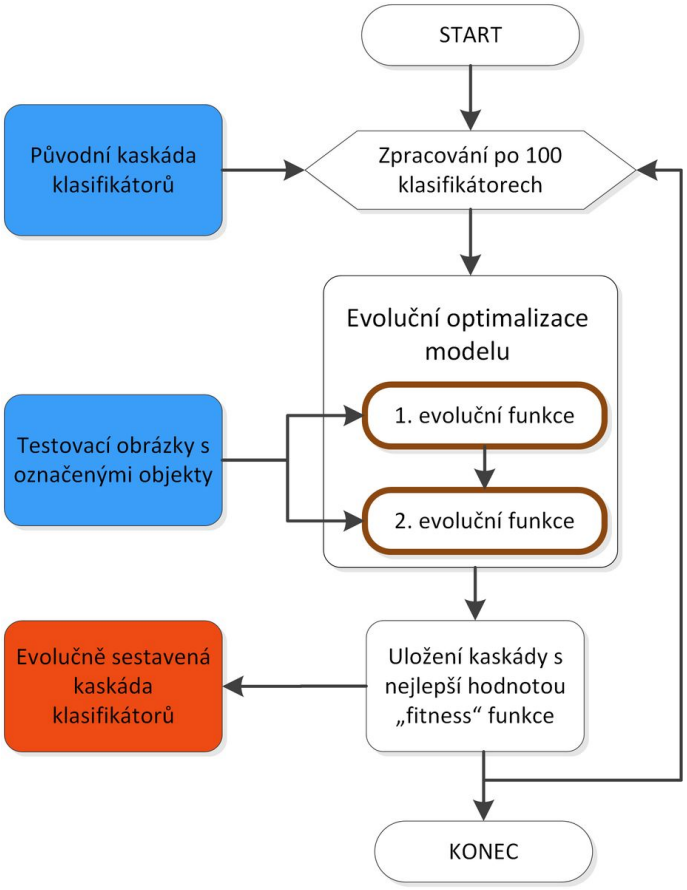
Input: slabé klasifikátory, integrální obrazy
Output: jeden nejlepší klasifikátor

1 transformace integrálních obrazů
2 transformace klasifikátorů
3 kopírování klasifikátorů a integrálních obrazů do paměti GPU
4 **foreach** *klasifikátor* **do**
5 | klasifikátor předpovídá třídu každého integrálního obrazu
6 | výpočet a nastavení chyby klasifikátoru
7 **end**
8 kopírování chyb všech klasifikátorů do CPU
9 nalezení klasifikátoru s nejnižší chybou
10 změna vah integrálních obrazů podle předpovězené třídy nejlepším klasifikátorem

Obr. 3.10: Schéma hledání nejlepších klasifikátorů algoritmem AdaBoost.

3.1.4 Evoluční optimalizace kaskádního modelu

V průběhu trénovacího procesu je vytvářen kaskádní klasifikátor. Parametry jakými jsou počet stupňů kaskády a počet klasifikátorů v jednotlivých stupních kaskády musí být určeny před začátkem trénovací fáze. Dalším parametrem, který je možné měnit, je prahová hodnota stupně kaskády. Určení těchto parametrů je složité, liší se případ od případu a vyžaduje již jisté zkušenosti. Při nevhodném určení těchto parametrů může být v případě velkého množství klasifikátorů v kaskádě (dojde k přetrénování) zvýšena výpočetní náročnost a snížena přesnost detekce. Na druhou stranu, pokud bude klasifikátorů menší množství, bude přesnost detekce snížena množstvím falešně positivních detekcí.



Obr. 3.11: Schéma evolučního skládání kaskády klasifikátorů.

Z těchto důvodů byly v rámci optimalizace kaskádního modelu (viz obrázek 3.11) implementovány evoluční algoritmy, jejichž cílem je rozřadit klasifikátory do stupňů a určit prahové hodnoty stupňů za účelem dosažení co nejlepší přesnosti na velkých testovacích snímcích. Pro vytvoření evolučních funkcí byla použita volně

dostupná knihovna JGAP[2] (Java Genetic Algorithms Package). Z důvodů výpočetní náročnosti evoluční optimalizace, kde tento proces trval v závislosti na řešeném problému i několik dní, byl tento algoritmus rozšířen o možnost být spuštěn ve výpočetním klastru čítajícím desítky výpočetních stanic (v tomto případě). Tento výpočetní klastr běžel na volně dostupném prostředí Apache Spark[3], které slouží pro distribuované výpočty. Výše zmíněné prostředí je vytvořeno v jazyku JAVA a funkcionálním jazyku SCALA. V samotné evoluční optimalizaci byla pro výpočet v klastru upravena (s použitím funkcionálního programování) nejvíce časově náročná „hodnoticí (fitness) funkce".

Tato evoluční optimalizace se týká pouze kaskády klasifikátorů. Kaskáda modelů umělé inteligence nemohla být evolučně optimalizována, protože tento proces byl na základě provedených výpočetních testů velmi výpočetně náročný. Problém byl, že při změně počtu klasifikátorů (příznaků) ve stupni kaskády, docházelo vždy k novému natrénování modelu na nové konfiguraci těchto příznaků, a až poté byl model aplikován na testovací snímky. V případě použití kaskády klasifikátorů byla evoluční optimalizace výpočetně méně náročná, protože po změně klasifikátorů ve stupních následovalo testování a vyhodnocení přesnosti na základě hodnoticí funkce.

Evoluční sestavování kaskády klasifikátorů

Schéma navrženého systému je patrné z obrázku 3.11. Vstupem funkce je kaskáda klasifikátorů natrénovaná algoritmem AdaBoost. Druhým vstupem jsou velké testovací obrázky spolu s popisem, kde se v každém obrázku nalézá hledaný objekt. Tento popis musí být vytvořen ručně, zvlášť pro každý obrázek. Evoluční trénování kaskády klasifikátorů probíhá v definovaných cyklech (v každém cyklu je přidáno 100 dalších klasifikátorů), kde v každém cyklu je optimalizována struktura kaskády klasifikátorů na základě evolučních funkcí a spočtena přesnost této kaskády na testovacích datech. Kaskáda klasifikátorů s nejvyšší přesností je nakonec výstupem této optimalizace. Vytvořené evoluční funkce využívají stejnou hodnoticí funkci pro ohodnocení sestavované kaskády klasifikátorů. Dále budou obě evoluční a hodnoticí funkce popsány.

Evoluční funkce

Evoluční optimalizace kaskády klasifikátorů je založena na dvou vytvořených evolučních funkcích. Nejprve je provedena optimalizace pomocí první funkce a poté je aplikována funkce druhá.

[2]Dostupný z URL: http://jgap.sourceforge.net/
[3]Dostupný z URL: http://spark.apache.org/

První funkce sestavuje kaskádu tak, že zkouší měnit celkový počet stupňů i počet klasifikátorů v každém stupni. Z pohledu evolučních algoritmů je vytvořen chromozom obsahující geny, kde každý gen reprezentuje jeden stupeň kaskády klasifikátorů. Každý gen dále obsahuje alely, jejichž hodnoty jsou v rozmezí od 0 až po maximální počet všech klasifikátorů v kaskádě. Pouze gen reprezentující první stupeň kaskády může nabývat hodnot od 1 do 3. Tento rozsah je omezen z důvodů, aby první stupeň kaskády pracoval co nejrychleji a zároveň zamítnul co největší množství negativních pod oken. V případě většího počtu klasifikátorů v prvním stupni by byl detekční proces velmi zpomalen. Po sestavení kaskády musí být pro každý stupeň určena prahová hodnota P podle rovnice:

$$P = \frac{1}{2} \sum_{t=1}^{T} \alpha_t, \tag{3.2}$$

kde α_t je váha klasifikátoru h_t.

Druhá evoluční funkce je aplikována na kaskádu po skončení první evoluční funkce a jejím účelem je měnit prahové hodnoty jednotlivých stupňů. Každý gen v chromozomu reprezentuje jeden stupeň kaskády a hodnoty alely každého genu jsou v rozsahu změny ($\pm 25\,\%$) z původní prahové hodnoty stupně.

Hodnoticí funkce

Tato hodnoticí (fitness) funkce ohodnocuje přesnost evolučně skládané kaskády klasifikátorů na testovacích snímcích. Principem je aplikování natrénované kaskády na všechny testovací snímky a detekování hledaných objektů v těchto snímcích, kde jsou souřadnice a rozměry takto detekovaných objektů porovnány s referenčními souřadnice, které jsou ručně vytvořeny pro každý testovací snímek. Způsob, jakým je ve vstupním snímku detekován objekt na základě aplikování kaskádního modelu, bude popsán v kapitole 3.1.6. Při výpočtu úspěšnosti je nutné uvažovat jistou toleranci při srovnání souřadnic detekovaného objektu a ručně definovaných souřadnic. Tato tolerance přesnosti byla definována v intervalu $\pm\frac{w}{2}$ pixelů, kde w je šířka detekovaného objektu v pixelech. Stejným způsobem je tolerance spočtena i pro výšku detekovaného objektu. Tato tolerance je relativně veliká, protože uživatelem označený střed a rozměry detekovaného objektu nemusí být přesné. Na základě těchto porovnání je spočtena hodnota hodnotící funkce C_{F} podle rovnice:

$$C_{\mathrm{F}} = \frac{1}{aN_{\mathrm{FN}} + bN_{\mathrm{FP}} + 0,1}, \tag{3.3}$$

kde N_{FN} určuje počet nedetekovaných objektů,N_{FP} určuje počet falešně positivních

detekcí. Koeficienty a a b jsou voleny experimentálně a určují váhy vztažené k proměnným N_{FN} a N_{FP}. Hodnota čísla $0,1$ ve jmenovateli zlomku zaručuje nedělitelnost nulou.

3.1.5 Použití hlubokého učení pro klasifikaci objektů

V rámci vytvořeného systému pro trénování a testování objektového detektoru (kaskádního modelu), bylo navrženo rozšíření pro klasifikaci objektů detekovaných tímto detektorem. Klasifikace je prováděna s pomocí moderních algoritmů hlubokého učení, konkrétně konvoluční neuronovou sítí. Cílem je na vstupní sadě různých kategorií obrazů (např. zdravá arterie, arterie obsahující plak) natrénovat model, který bude následně aplikován na objekt detekovaný kaskádním modelem. Zapojení tohoto natrénovaného modelu do celého detekčního systému bude podrobně popsáno v kapitole 3.1.6. Nyní bude popsáno použité prostředí pro hluboké učení.

Prostředí pro hluboké učení

V současné době je téma hlubokého učení velmi rozšířený pojem, a proto existuje celá řada volně dostupných stále se vyvíjejících prostředí pro hluboké učení. Většina těchto prostředí dokáže k výraznému urychlení výpočtu používat paralelní hardware, jakým jsou například grafické akcelerátory. Mezi současné prostředí pro hluboké učení patří:

- DL4J - Deeplearning for Java (*http://deeplearning4j.org/*),
- BVLC/Caffe (*http://caffe.berkeleyvision.org/*),
- Theano (*http://www.deeplearning.net/software/theano/*),
- Torch (*http://torch.ch/*),
- Tensor Flow (*https://www.tensorflow.org/*),
- cuda-convnet (*https://github.com/akrizhevsky/cuda-convnet2*).

V rámci této práce bylo zvoleno prostředí DL4J zejména protože je tento systém vytvořen v programovacím jazyce JAVA a dále, protože je možné výpočty distribuovat do výpočetního klastru. Toto prostředí je dále založeno na již zmíněném distribuované prostředí Apache Spark a platformě Hadoop[4], která slouží také k distribuování dat do klastru. Prostředí DL4J umí pracovat s různými typy dat (obraz, audio, video, strukturovaná data). Dále podporuje celou řadu algoritmů hlubokého učení, jakými například jsou:

- konvoluční sítě,

[4]Dostupný z URL: http://hadoop.apache.org/

- rekurentní sítě,
- autoenkodéry
- *deep-belief* sítě.

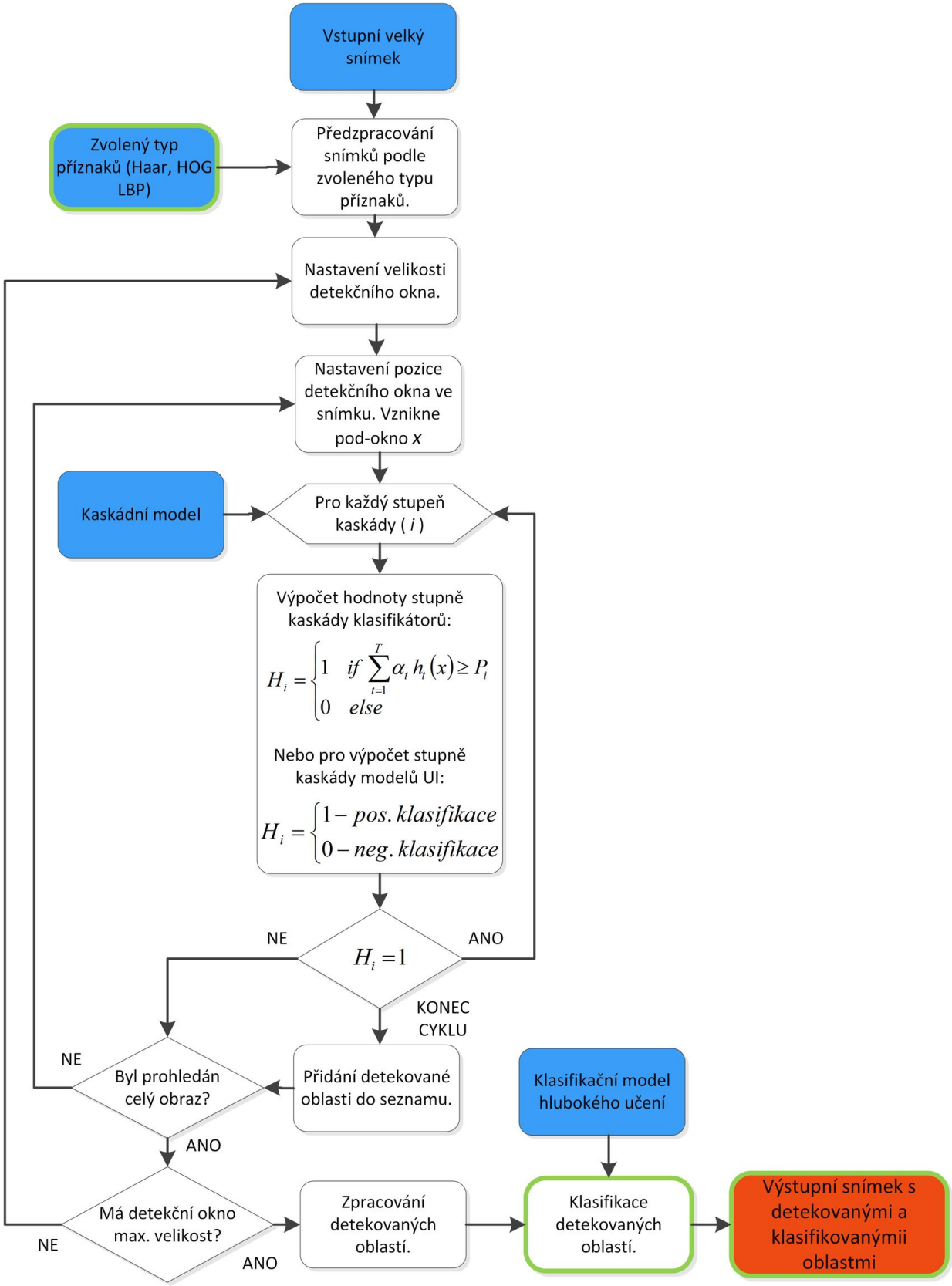
Na základě zvoleného algoritmu a nastavení parametrů této sítě je natrénován model, který je pak vstupem detekčního procesu.

3.1.6 Detekční a klasifikační proces

Detekční proces slouží k aplikování natrénovaného kaskádního modelu na vstupní obraz. Klasifikační proces dále klasifikuje výstupní detekované objekty. Princip celého procesu je patrný z obrázku 3.12. Vstupem procesu je snímek, ve kterém budou hledány a klasifikovány objekty. Dalšími vstupy jsou kaskádní a klasifikační model. Celý proces začíná předzpracováním vstupního snímku podle zvoleného typu obrazových příznaků, dále jsou nastaveny startovní rozměry detekčního okna, parametr velikosti zvětšení detekčního okna po každé iteraci a parametr velikosti posunu detekčního okna ve vstupním snímku (definováno jako součin velikosti detekčního okna a hodnoty velikosti posunu). Po nastavení startovní velikosti detekčního okna je nastavena pozice detekčního okna ve snímku. Tímto vznikne výřez ze vstupního snímku (pod okno). Toto pod okno je dále posláno na první stupeň kaskádního modelu, pokud je vyhodnoceno jako positivní, je dále zpracováno následujícím stupněm, v opačném případě už dále není toto pod okno zpracováváno a proces pokračuje vytvořením dalšího pod okna. V případě, že pod okno projde všemi stupni kaskádního modelu, je přidáno do seznamu detekovaných oblastí. Dále je detekční okno posouváno s definovaným krokem, dokud se neposune přes celý snímek. Pak následuje zvětšení velikosti detekčního okna a celý proces se opakuje, dokud není velikost detekčního okna větší než menší ze stran vstupního obrazu.

Zpracování detekcí

Poté, co skončí detekční proces, jsou všechny uložené detekované oblasti dodatečně zpracovány, aby výstupem nebylo velké množství detekcí (každý objekt bývá většinou detekován vícenásobně), ale je vyhodnocena pouze nejvíce vypovídající detekce daného objektu. Zpracování detekcí probíhá následovně: nejprve jsou hledány podobné detekce jedné oblasti (buď se nachází jedna detekce uvnitř jiné, nebo se překrývají o 20 % šířky jedné z detekcí), dále je určen jejich počet pro každou oblast a pokud je tento počet větší než je nastavená hodnota (kvůli omezení falešně positivních detekcí), tak jsou detekce v rámci každé oblasti zvlášť zprůměrovány v jednu výstupní, nejvíce vypovídající detekci pro každou oblast.



Obr. 3.12: Schéma detekčního a klasifikačního procesu.

Klasifikace detekovaného objektu

Takto detekované objekty ve vstupním obrazu mohou být dále klasifikovány modelem hlubokého učení. Výstupem klasifikace je detekce označená jménem kategorie snímku, která byla klasifikována s největší jistotou. Pokud je výstupem detekčního procesu falešně positivní snímek, který klasifikační model nezná, je možné nastavit prahovou hodnotu pro klasifikaci, aby v případě takovýchto falešně positivních detekcí byla klasifikační kategorie označena jako neznámá. Klasifikace detekovaných objektů byla v této práci použita např. při klasifikování stupně plaku v artérii.

3.2 Optimalizace učících se algoritmů umělé inteligence

Optimalizace učících se algoritmů umělé inteligence spočívá zejména v možnostech spouštět tyto algoritmy na paralelním hardware, jakými jsou více procesorové systémy, grafické akcelerátory nebo výpočetní klastr. Dále pak spočívá ve způsobu samotné implementace, kdy použitím nízko úrovňového, neobjektového programování a vhodných implementačních technik pracuje výsledný algoritmus efektivněji (většinou na úkor jednoduché rozšiřitelnosti a úpravy algoritmu). V rámci učících se algoritmů je cílem optimalizovat zejména trénovací proces, který je výpočetně mnohem více náročnější než samotný testovací proces. Obecně lze říci, že u většiny algoritmů není možné paralelizovat celý trénovací proces (jedná se o sekvenční části algoritmu), většinou jde provést pouze dílčí paralelizaci. Na druhou stranu pro testovací proces je paralelizace možná vždy. Při srovnání vývoje paralelních algoritmů na více jádrovém CPU a grafických akcelerátorech je vývoj pro GPU značně ztížen vlivem limitovaného množství instrukcí dostupných pro architekturu SIMT, která pracuje s velkým množstvím velmi jednoduchých výpočetních jader, na rozdíl od architektury SIMD (Single Instruction Multiple Data - pro CPU), kde je pouze několik výpočetních jader, která ale dokáží pracovat s velkým množstvím instrukcí, což usnadňuje vývoj algoritmů.

V rámci práce byly pro více jádrová CPU paralelizovány testovací části algoritmů: rozhodovací strom, náhodný les, systémy podpůrných vektorů (SVM), neuronová síť, k-NN. Algoritmy SVM, náhodný les a neuronová síť byly převzaty pod volnou licencí z jiných projektů a následně upraveny. Trénovací část byla alespoň částečně paralelizována pro všechny výše zmíněné algoritmy kromě SVM, které je v dané implementaci navrženo celé sekvenčně. Tyto algoritmy je možné kombinovat se systémem pro detekování objektů v obrazech nebo s algoritmem trénovatelné segmentace.

Vzhledem ke komplikovanosti vývoje algoritmů pro grafické akcelerátory byl optimalizován pouze algoritmus k-NN, který je vhodný pro řešení jednodušších problémů, na druhou stranu je schopen zpracovávat velké objemy dat, čehož běžné neparalelizované algoritmy nejsou v takovém rozsahu schopné. V případě řešení složitějších problémů (hůře separovatelná data) je nutné využít jiné algoritmy (např. SVM). Další nespornou výhodou využití grafických akcelerátorů pro výpočet je spotřeba elektrické energie, která byla pro vybraný řešený problém desetkrát nižší, než v případě použití klasických CPU systémů. V případě pořizovací ceny je tato cena v současné době řádově nižší pro GPU akcelerátory, ve srovnání s CPU systémem, který má stejný výpočetní výkon.

Dále budou popsány platformy pro vývoj algoritmů schopných běžet na grafických akcelerátorech. Optimalizovaný algoritmus k-NN musel být upraven oproti klasické procesorové verzi, aby dokázal vhodně využít architekturu GPU. Popis tohoto algoritmu bude uveden níže.

3.2.1 Platformy CUDA a OpenCL

V dnešní době existují dvě nejpoužívanější platformy podporující výpočty na grafických akcelerátorech. Jedná se o platformy CUDA [29] a OpenCL [92]. Platforma CUDA je sice více používaná, ale na druhou stranu je omezená pro použití pouze grafických akcelerátorů od společnosti NVIDIA. Platforma OpenCL je využívána méně, nicméně tato platforma může být použita různými typy zařízení. To znamená, že vytvořený program může být spuštěn na zařízeních všech výrobců podporující standard OpenCL, což se týká grafických akcelerátorů, klasických CPU i dalších speciálních zařízení.

Při srovnání architektur GPU s CPU, je architektura GPU více specializována pro intenzivní vysoce paralelní výpočty. Obrázek 3.13 zobrazuje rozdíly mezi jednotlivými architekturami, kde GPU architektura obsahuje v dnešní době až tisíce výpočetních jednotek (ALU - aritmeticko-logická jednotka) v porovnání s běžnou CPU architekturou obsahující pouze jednotky těchto výpočetních jednotek, které mohou pracovat v jeden okamžik současně.

OpenCL je otevřená platforma, která vychází z principu platformy CUDA (bylo již podrobněji popsáno v kapitole 3.1.3 - Prostředí CUDA). V praxi se prostředí liší hlavně dostupnými funkcemi při programování výpočetního jádra aplikace (kernel) a dále názvy jednotlivých logických částí viz tabulka 3.1. OpenCL je založeno na jazyce C, proto bylo pro vzájemnou kooperaci mezi jazyky C a JAVA použito rozhraní jocl[5].

[5]Dostupné z URL: `http://www.jocl.org/`

(a) Schéma CPU architektury (b) Schéma GPU architektury

Obr. 3.13: Rozdíl mezi architekturami CPU a GPU

3.2.2 Optimalizovaný algoritmus *k*–nejbližších sousedů

Prvně byl pro zpracování velkých databází použit algoritmus k–NN z volně dostupného programu RapidMiner[6], který patří v současné době mezi nejpoužívanější prostředí pro analýzu dat. Tato verze CPU algoritmu byla bohužel velmi pomalá. Z tohoto důvodu byl vytvořen vlastní optimalizovaný GPU algoritmus. Vytvořená verze pracuje s mnoha metrikami (např. Manhattanská, Euklidovská, kosinová, . . .).

Princip algoritmu je vysvětlen ve schématu na obrázku 3.14. Vstupem algoritmu jsou trénovací a testovací databáze, které jsou převedeny do datových polí o formátu *float4* [92], který je schopen v jednom kroku zpracovat 4 hodnoty typu *float*. Dále je kernel optimalizován například použitím rychlých lokálních pamětí. Logika algo-

[6]Dostupné z URL: `https://rapidminer.com/`

Tab. 3.1: Vzájemná terminologie prostředí CUDA a OpenCL.

CUDA		OpenCL	
mřížka	(Grid)	rozsah	(NDRange)
blok vláken	(Thread Block)	pracovní skupina	(Work–group)
vlákno	(Thread)	pracovní prvek	(Work item)
ID vlákna	(Thread ID)	globální ID	(Global ID)
index bloku	(Block index)	ID bloku	(Block ID)
index vlákna	(Thread index)	lokální ID	(Local ID)
sdílená paměť	(Shared Memory)	lokální paměť	(Local Memory)
lokální paměť	(Local Memory)	privátní paměť	(Private Memory)

ritmu k–NN musela být oproti běžně používané verzi algoritmu modifikována kvůli efektivnímu nasazení na GPU. V původní CPU verzi jsou vždy spočteny všechny vzdálenosti mezi testovacím prvkem a všemi prvky trénovacími a až nakonec je hledáno k nejbližších sousedů. Z důvodů obrovské paměťové náročnosti (každé paralelní vlákno si musí alokovat pole pro uložení všech vzdáleností) byl algoritmus modifikován tak, aby k nejbližších sousedů bylo hledáno po každém spočtení vzdálenosti mezi testovacím a trénovacím prvkem. Dále je sečten počet sousedů pro každou třídu a následně vybrána třída s největším počtem těchto sousedů, která je nastavena jako predikovaná třída pro daný testovací prvek. Vytvořený algoritmus byl publikován v pracích autora [3, 4].

Input: trénovací data (*float4*), testovací data (*float4*)
Output: predikované hodnoty

```
1  načtení testovacího prvku z databáze
2  foreach trénovací prvek do
3      foreach atribut do
4          výpočet vzdáleností mezi atributy trénovacího a testovacího prvku;
           suma vzdáleností
5      end
6      for k = 0 to počet nejbližších sousedů do
7          if suma vzdáleností < vzdálenost pro k–tého souseda then
8              vzdálenost k–tého souseda = suma vzdáleností
9              posunutí hodnot vzdáleností pro další k–té vzdálenosti
10             break
11         end
12     end
13 end
14 čítání počtů sousedů pro každou třídu
15 vybrání třídy s nejvyšší počtem sousedů
16 nastavení predikce podle třídy
```

Obr. 3.14: Schéma optimalizovaného algoritmu k–nejbližších sousedů.

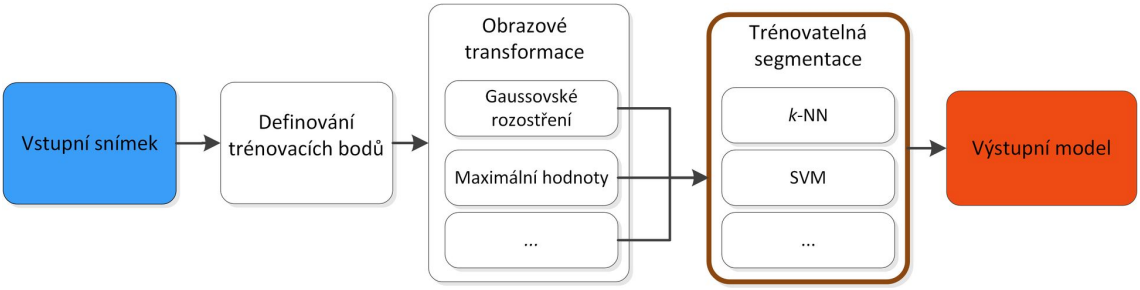
Podpora pro více grafických akcelerátorů

Pro paralelní běh GPU algoritmů na větším počtu grafických akcelerátorů současně byla v rámci práce vytvořena knihovna, která dokáže automaticky a optimálně roz-

dělit činnost mezi všechny dostupné grafické akcelerátory. Vstupní data pro GPU jsou rozdělena a kopírována do příslušných typů pamětí podle toho, jakého jsou typu a jestli musí být dostupná na všech zařízeních nebo jestli pouze musí být dílčí část dat na jednom zařízení. Tato knihovna je založena na platformě OpenCL a jednou z jejích výhod je jednoduché a rychlé použití při vytváření nových GPU algoritmů.

3.3 Metoda trénovatelné segmentace

V rámci této kapitoly bude popsána rychlostní a paměťová optimalizace testovacího procesu trénovatelné segmentace, jejíž princip byl již vysvětlen v kapitole 1.4. Při zpracování obrazů s vyšším rozlišením (např. letecké snímky) je v případě velkého množství zpracovávaných dat nutné dosáhnout přijatelného času jejich zpracování. Proto byla vytvořena verze optimalizovaná pro více procesorové systémy.



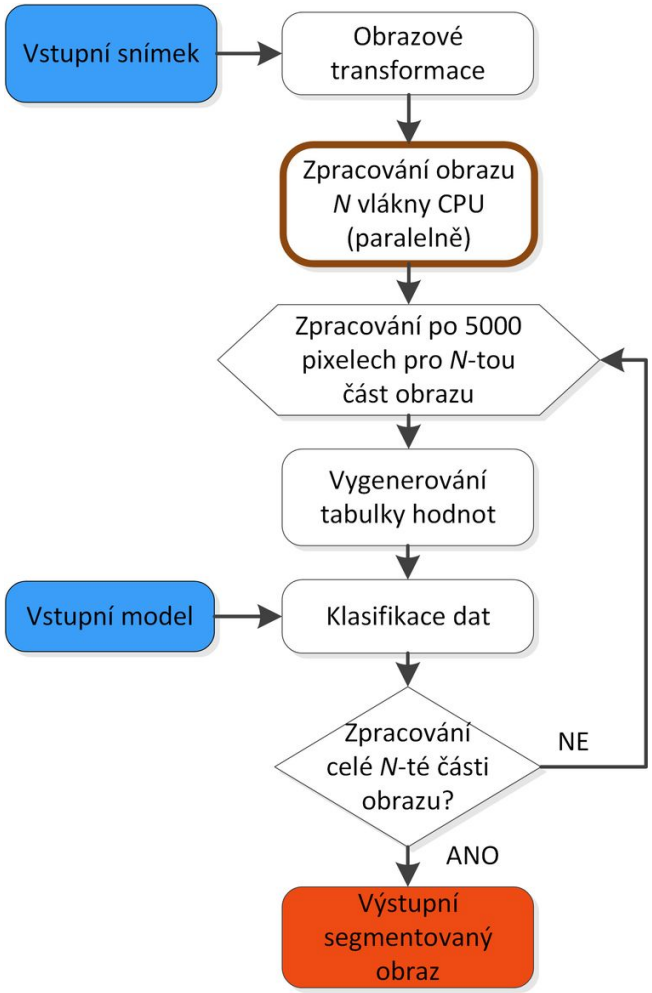
Obr. 3.15: Navržený trénovací proces trénovatelné segmentace.

Optimalizace trénovací části (viz obrázek 3.15) procesu trénovatelné segmentace nebyla provedena, protože trénování probíhá zpravidla na souboru dat obsahujícím maximálně stovky hodnot ručně definovaných uživatelem. Jedná se tedy o výpočetně nenáročný proces. Výstupní natrénovaný model je použit v následném testovacím procesu.

3.3.1 Optimalizace testovací části

Vstupem optimalizovaného testovacího procesu (viz obrázek 3.16) je snímek určený k segmentaci spolu s klasifikačním modelem. Na snímek jsou aplikovány stejné obrazové transformace, jako tomu bylo v případě trénování modelu. Dále následuje optimalizovaná část testovacího procesu: vstupní snímek je nejprve rozdělen na N částí, kde N značí počet použitých paralelních vláken CPU. Znamená to tedy, že všechny části obrazu jsou zpracovávány navzájem paralelně. Dále je každá N-tá část obrazu zpracována následovně: její zpracování probíhá ve smyčce po 5000 pixelech (zvoleno experimentálně), kdy jsou nejprve na základě již dříve transformovaných

obrazů vyčteny hodnoty všech obrazů pro danou pozici pixelu a uloženy do jednoho řádku tabulky, na kterou je dále aplikován natrénovaný model, který predikuje hodnoty jednotlivých výstupních pixelů. Po skončení výpočtu všech CPU vláken je výstupem segmentovaný obraz.



Obr. 3.16: Navržený paralelní testovací proces trénovatelné segmentace.

V rámci optimalizace tohoto procesu proběhlo experimentální měření ohledně počtu pixelů obrazu zpracovávaných v jedné dávce. Hodnota 5000 vyšla jako vhodný kandidát oproti původním možnostem, kdy se zpracovával pouze pixel po pixelu (časově náročné) nebo v případě, kdy byla celá část obrazu zpracována najednou, což bylo zase paměťově náročné (zejména při vyšší míře paralelizace).

4 OVĚŘENÍ NOVÝCH METOD NA VYBRANÝCH PŘÍPADECH

Funkčnost vytvořených algoritmů byla ověřena na vybraných příkladech. Trénovatelná segmentace byla použita pro řešení problému detekce domů ze satelitních snímků a pro segmentaci obrazových dat meteoritů. Optimalizovaný algoritmus k-nejbližších sousedů byl nasazen pro klasifikaci velkých objemů dat. Systém pro detekci objektů v obrazech byl použit pro trénování optimalizovaného obličejového detektoru a dále pro lokalizaci arterie z ultrazvukového snímku a klasifikaci úrovně jejího plaku. Dosažené výsledky byly nakonec porovnány s výsledky jiných přístupů a byl diskutován vlastní přínos.

Výpočetní prostředí

Všechny experimenty (nebude-li uvedena jiná konfigurace pro dané měření) popsané v této kapitole probíhaly s použitím osobního počítače s procesorem Intel Core i7–3770 s frekvencí 4,1 GHz, 32 GB pamětmi RAM a dvěma dvou-jádrovými grafickými akcelerátory GeForce GTX 690 [93]. Každé GPU se skládá ze dvou jader, kde má každé 1536 výpočetních CUDA jednotek. Velikost pamětí byla 4096 MB s frekvencí 6 GHz. Teoretický výkon jedné karty byl v jednoduché přesnosti 5,62 TFLOPS. Celková výkonnost obou akcelerátorů byla tedy 11,24 TFLOPS. Při plném vytížení jednoho GPU byl naměřen příkon 300 W. Při plném vytížení výpočetní stanice bez pracujících grafických akcelerátorů byl naměřen příkon 180 W.

4.1 Detekce domů ze satelitních snímků s pomocí trénovatelné segmentace

Pro lokalizaci zeměpisných souřadnic domů ze satelitních snímků byla použita metoda trénovatelné segmentace. Tato metoda byla publikována v práci [6]. Výsledky detekce domů ze satelitních snímků byly také publikovány v článcích [8] a [88].

4.1.1 Trénování klasifikačního modelu

Trénovací proces načítá vstupní snímek, ve kterém jsou uživatelem zelenými body označena místa (viz obrázek 4.1a), kde se nachází domy a červenými body je označeno okolí domů. Dále byly provedeny obrazové transformace. Jednou transformací je saturace obrazu (viz obrázek 4.1b) a druhou transformací je extrakce hodnot odstínů spolu s následným Gaussovským rozostřením. Dále jsou získány hodnoty

pixelů originálního a transformovaných obrazů v definovaných bodech (zelených a červených) a vzniklý data set je vstupem učícího se algoritmu umělé inteligence. V tomto případě byl experimentálně zvolen algoritmus náhodného lesa. Výstupem procesu je natrénovaný klasifikační model.

(a) Vybrané trénovací body. (b) Saturovaný vstupní obraz. (c) Extrakce hodnot odstínu.

Obr. 4.1: Trénovací obraz a jeho transformace.

4.1.2 Dosažené výsledky

Pro ověření funkčnosti detekce domů bylo vybráno 5 snímků evropských hlavních měst. Dále byl každý snímek rozdělen na dvě poloviny - trénovací a testovací a následně byl pro každé město natrénován vlastní model. K testovacím částem snímků byly ručně vytvořeny masky označující pozice domů. Proces detekce domů ze snímku Berlína je zobrazen na obrázku 4.2, kde vstupní snímek (viz obrázek 4.2a) je segmentován (viz obrázek 4.2b) a následně prahován a filtrován mediánovým filtrem (viz obrázek 4.2c). Tento výstup je možné převést do geografického formátu *kml* vyvinutého společností Google a následně zobrazit v aplikaci Google earth (viz obrázek 4.3).

Pro zjištění přesnosti detekce A_{CC} byly srovnány masky s detekovanými oblastmi. Výsledky detekcí jsou zobrazeny v tabulce 4.1, kde F_{NR} označuje falešně negativní míru a F_{PR} označuje falešně positivní míru. Průměrná dosažená přesnost byla 78.3 %, kdy v některých případech je i pro člověka obtížné určit, jestli se jedná o střechu domu nebo jiný objekt. Dále v některých případech měla střecha domu stejnou barvu a tvar jako okolí (například šedivá silnice), což snižovalo přesnost klasifikace.

Pokud je model natrénovaný na snímku Paříže použit pro klasifikaci snímku Berlína, dosažená přesnost je 72,2 %. Je to způsobeno jinými tvary i barvou střech budov v Paříži a Berlíně.

4.1.3 Srovnání s jinými pracemi

Detekce domů z leteckých snímků byla řešena i jinými autory v následnicích pracech: v [94] je zvolena metoda segmentace v kombinaci s vektorově orientovaným detektorem hran, která dosahovala na snímcích s vysokým počtem budov rozličných tvarů přesnosti detekce 73 %. V práci [95] byla použita metoda, při níž byly extrahovány příznaky zjišťující informace o hranách a pozici stínu domu. Tento systém dosáhl přesnosti 95 %. V tabulce 4.2 jsou uvedeny výsledky přesností detekcí, kde vytvořený postup dosáhl 78 % přesnosti. Uvedené metody byly testovány na různých

(a) Originální testovací snímek (b) Snímek po aplikování trénovatelné segmentace. (c) Prahovaný snímek.

Obr. 4.2: Výsledky testovacího procesu.

Obr. 4.3: Vizualizace výsledků programem Google earth - Berlín.

Tab. 4.1: Přesnost klasifikace domů pro jednotlivá města.

City	A_{CC} [%]	F_{NR} [%]	F_{PR} [%]
Berlín	79,2	18,7	24,3
Budapešť	76,7	21,4	29,7
Lisabon	**80,3**	21,5	18,2
Paříž	76,1	40,1	12,6
Varšava	79,2	21,6	18,6

databázích leteckých snímků a proto je jejich srovnání spíše informativní. Předností nového postupu je pak rychlost vyhodnocení oproti předchozím metodám.

Tab. 4.2: Porovnání jednotlivých řešení pro detekci domů z leteckých snímků.

Řešení	A_{CC} [%]	Metoda
Ok (2008) [94]	73	Segmentace & detekce hran
Sirmacek a kol. (2008) [95]	95	Detekce hran a stínů
Nový postup	78	Trénovatelná segmentace

4.2 Segmentace obrazových dat meteoritů

Díky počítačové tomografii (CT - Computed Tomography - metoda využívající rentgenové záření) je v dnešní době možné zobrazovat například vnitřní orgány v tělech živočichů nebo lze CT využít pro odlišení materiálů obsažených v různých tělesech. Výstupem metody je série obrázků snímaného objektu tvořící dohromady 3D obraz. Pro následné oddělení jednotlivých částí 3D obrazu od sebe jsou využívány segmentační metody, díky kterým jsou tyto různé části materiálů lokalizovány, dále mohou být určeny jejich rozměry nebo spočten objem.

V rámci práce byl řešen problém segmentace meteoritů a materiálů obsažený v meteoritech. Zkoumané 3 meteority byly nalezeny v různém časovém období v blízkosti Žďáru nad Sázavou. Snímkování meteoritů probíhalo ve spolupráci s měřící laboratoří Středoevropského technologického institutu (CEITEC - Central European Institute of Technology). Cílem bylo ze získaných 3D obrazů segmentovat meteorit od jeho pozadí, dále segmentovat jednotlivé materiály (železo, nikl, troilit) obsažené v meteoritu a určit jejich množství. Naměřené výsledky byly odeslány pro publikování do mezinárodního konferenčního sborníku [9].

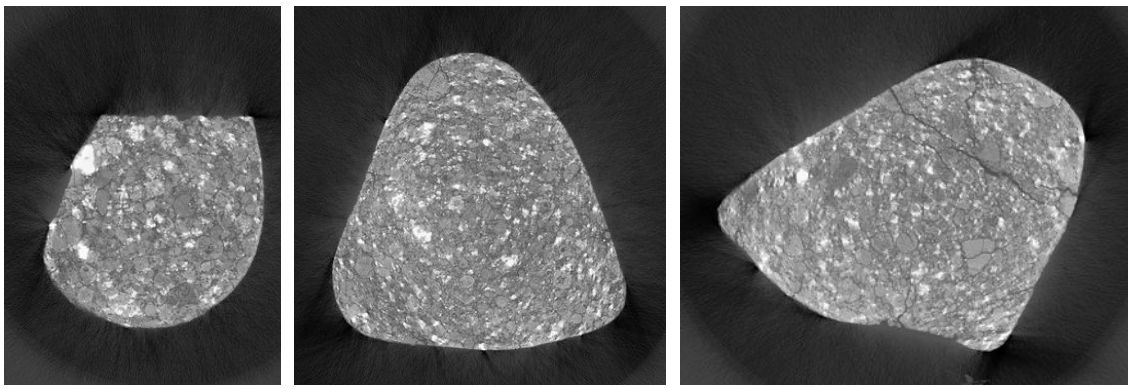
4.2.1 Použitá data

Všechna data byla nasnímána na tomografickém přístroji GE phoenix v|tome|x L240 (General Electric, USA), vybaveným rentgenovou trubicí s maximálním výkonem 180 kV/15 W a detektorem DXR250 s rozlišením 2048x2048 pixelů při velikosti pixelu 200x200 μm^2. Parametry měření jednotlivých meteoritů jsou uvedeny v tabulce 4.3.

Tab. 4.3: Parametry nastavení CT pro jednotlivé meteority.

	Spouštěcí napětí [kV]	Proud v rentg. trubici [μA]	Expoziční čas [ms]	Počet projekcí	Velikost voxelu [μm]
Meteorit 1	150	110	500	2200	15
Meteorit 2	160	150	500	2200	24
Meteorit 3	150	190	750	2200	28

Pro každý meteorit bylo tedy nasnímáno 2200 snímků. Vlivem různé velikosti jednotlivých meteoritů a směru snímání, byly snímky neobsahující data meteoritů smazány. Po této redukci obsahovaly 3D obrazy meteoritů postupně 1518, 1500 a 1364 snímků. Pro lepší vizuální čitelnost dat musela být provedena ekvalizace histogramu. Ukázky vybraných řezů jednotlivých meteoritů jsou zobrazeny na obrázku 4.4.



(a) 1. meteorit (b) 2. meteorit (c) 3. meteorit

Obr. 4.4: Ukázka dat meteoritů.

4.2.2 Navržené řešení

Prvním cílem bylo oddělit pozadí od meteoritu. K tomu měl být podle prvotních odhadů použit jednoduchý algoritmus pro prahování obrazu. Bohužel zvolená prahovací hodnota, která fungovala pro jednu část snímků 3D obrazu, nefungovala správně již

pro jinou část. Problém byl v kolísající světelné intenzitě jednotlivých snímků, kde maximální rozdíl mezi průměrnou pixelovou hodnotou dvou obrazů byl 60 (v rozsahu 0-255). Proto, aby byly správně segmentovány jednotlivé materiály obsažené v meteoritu, musel být navržen normalizační algoritmus, který před zpracovával snímky určené k segmentaci.

V této kapitole bude popsána segmentace meteoritu od pozadí, dále normalizační proces a segmentace jednotlivých materiálů. Pro segmentaci materiálů byl vždy nejprve vyzkoušen algoritmus prahování, ale protože v některých případech nelze rozpoznat jednotlivé materiály jen podle barvy, ale i podle tvaru, umístění, atd., musel být vytvořen složitější proces obsahující trénovatelnou segmentaci zpravidla s dalšími metodami pro zpracování obrazu.

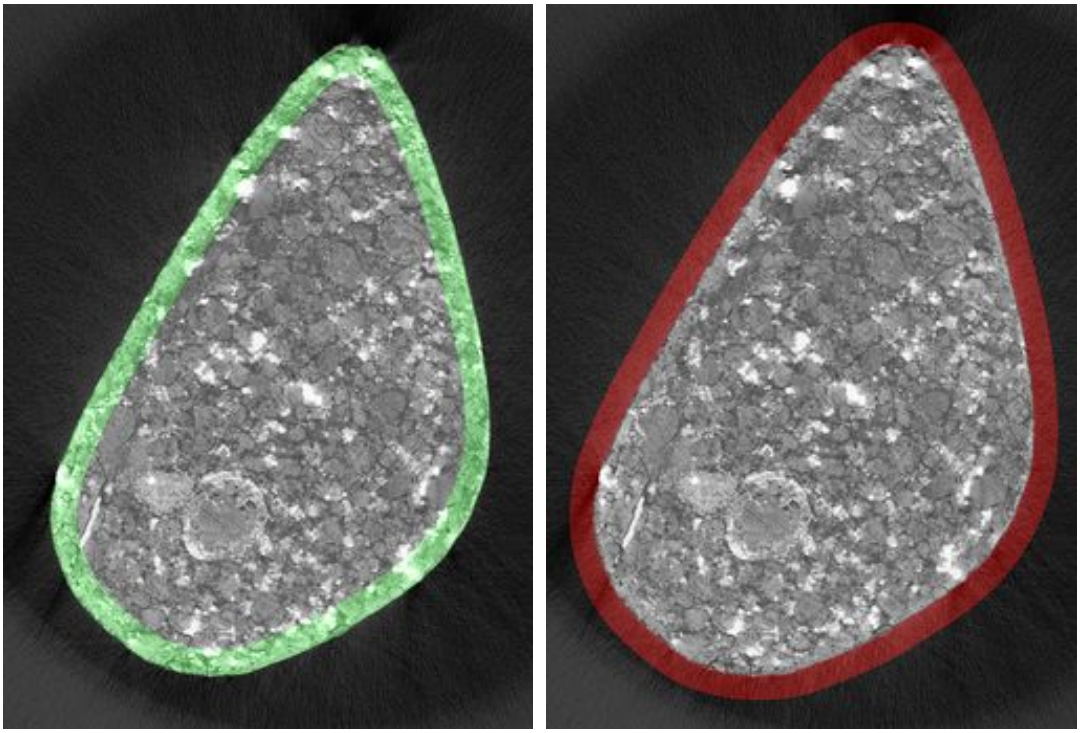
Segmentace meteoritu

Jak již bylo zmíněno, jednoduché prahování snímků za účelem oddělení meteoritu od pozadí, nebylo dostatečné. Proto byl zvolen komplexnější postup. Nejprve bylo náhodně vybráno 500 snímků třetího meteoritu, které byly prahovány. Tím byla zvlášť označena část meteoritu a zvlášť část pozadí snímku. Nastavení prahovací hodnoty se měnilo manuálně v závislosti se zvoleným snímkem. Tyto snímky byly vstupem procesu, jehož účelem bylo v první fázi vygenerovat pro každý snímek positivní a negativní trénovací body v oblasti přechodu (nejvyšší výskyt nepřesností) mezi meteoritem a pozadím.

Vstupní snímek označující pozici meteoritu byl podroben detekci hran. Tím byl získán obrys meteoritu. Šířka obrysové čáry byla zvětšena operacemi Gaussovským rozostřením s poloměrem 25 a následným prahováním s hodnotou 1. Pro získání oblasti okrajů meteoritu byl původní obraz zobrazující meteorit odečten od nově vzniklého obrysu meteoritu a výsledná oblast je zobrazena zeleně v obrázku 4.5a. Pro získání oblasti přechodu (viz obrázek 4.5b) mezi meteoritem a pozadím byl obrys meteoritu invertován a odečten od obrazu původního.

V další fázi procesu byly trénovací body a 500 snímků meteoritu použity v trénovacím procesu trénovatelné segmentace, kde byly vstupní snímky spolu se zvolenými obrazovými příznaky (prostorové - scale space, Gaussovské, maximální hodnoty a hodnoty rozptylu), použity pro vygenerování trénovacího setu, který byl vstupem experimentálně zvoleného učícího se algoritmu náhodného lesa. Algoritmus měl nastaven parametr maximální hloubky na 30 a počet vytvářených stromů na 10. Před natrénováním modelu umělé inteligence byl vytvořený trénovací set vyvážen tak, aby počty trénovacích positivních a negativních prvků byly stejné. Výstupní model byl uložen pro použití v testovacím procesu trénovatelné segmentace.

Pro segmentaci meteoritu od pozadí, která byla provedena na všech zbývajících



(a) Oblast pro generování positivních bodů. (b) Oblast pro generování negativních bodů.

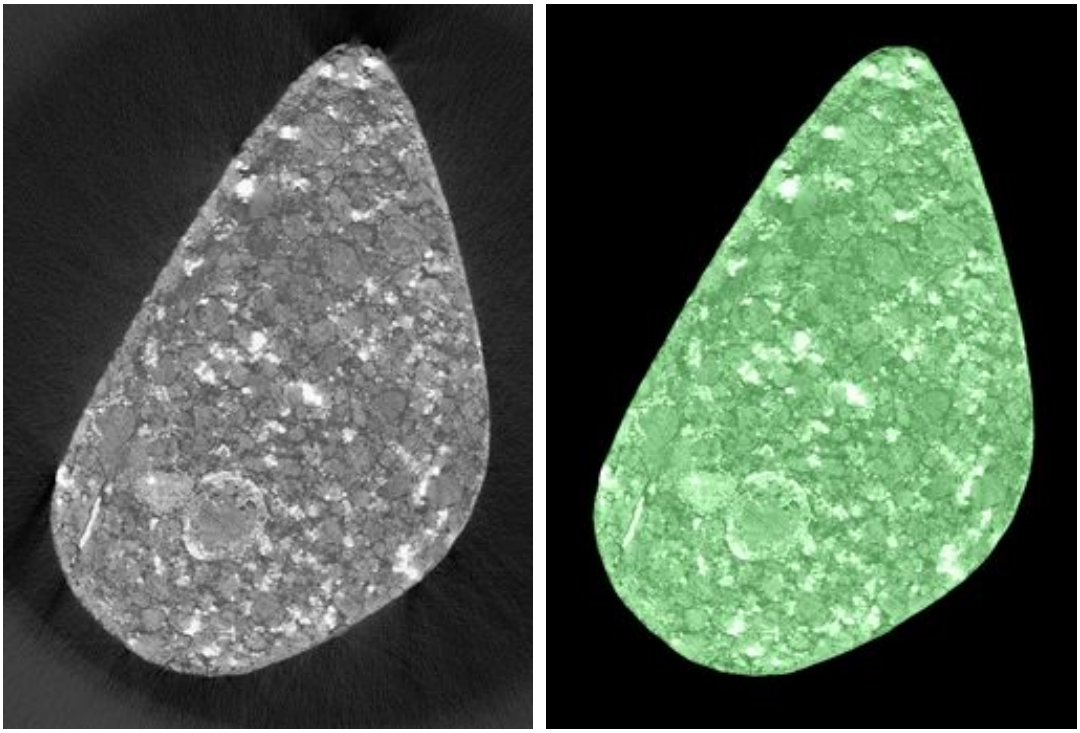
Obr. 4.5: Vyznačené oblasti pro generování trénovací bodů.

snímcích, byl vytvořen proces skládající se z trénovatelné segmentace a dalších metod pro finální zpracování snímků. Na výstupní segmentovaný obraz byl dále aplikován algoritmus Gaussovského rozostření s poloměrem 3 a algoritmus pro automatické vyplnění prázdných míst (fill holes). Výstupem byl binární obraz zobrazující meteorit bíle a pozadí černě. Po aplikování binárního obrazu na vstupní snímek (viz obrázek 4.6a) je lokalizován meteorit (viz obrázek 4.6b).

Normalizace snímků

Pro normalizaci snímků bylo předpokládáno, že jednotlivé složky materiálů meteoritu jsou v průměru zastoupeny ve stejné míře ve všech snímcích. Navržená normalizace snímků probíhala ve dvou fázích. Nejprve byly původní snímky spolu s označenou částí meteoritu použity pro získání celkové průměrné hodnoty pixelů a poté byla na základě této hodnoty provedena normalizace všech snímků. Průměrná pixelová hodnota byla získána jako průměr hodnot všech segmentovaných snímků meteoritů a jejich pozadí. Pro meteority byla spočtena celková průměrná hodnota pixelů 155 a dále pro jejich pozadí byla celková průměrná hodnota pixelů 36. Výsledná průměrná hodnota byla tedy 96.

Při následné normalizaci byly pro každý snímek opět spočteny průměrné hod-



(a) Originální snímek (b) Segmentovaný meteorit

Obr. 4.6: Ukázka segmentace meteoritu.

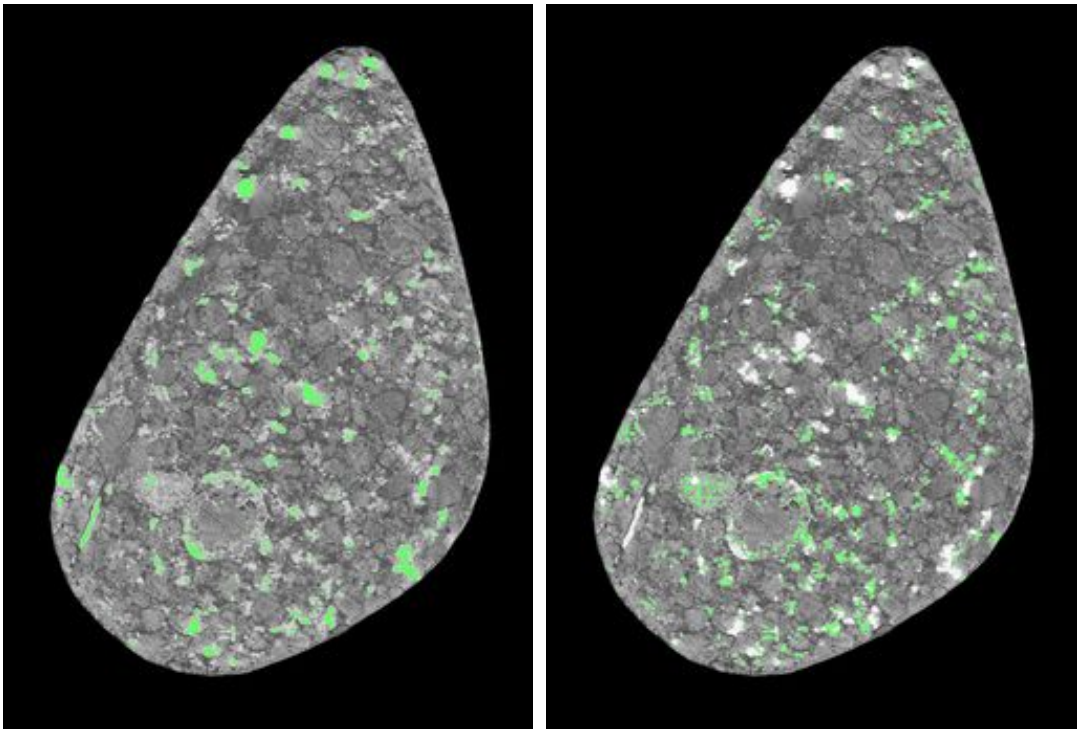
noty částí meteoritu a jeho pozadí. Vzniklý průměr těchto dvou hodnot byl srovnán s výslednou průměrnou hodnotou 96 a rozdíl těchto dvou průměrných hodnot byl poté přičten nebo odečten od jednotlivých hodnot pixelů normalizovaného snímku. U takto normalizovaných snímků bylo při segmentaci jednotlivých materiálů dosaženo větší přesnosti.

Segmentace železa a niklu

Pro segmentaci prvků železa (Fe) a niklu (Ni) byl použit pouze jednoduchý prahovací algoritmus s prahovou hodnotou 186. Pro oba tyto kovy byl použit stejný segmentační algoritmus, protože mezi nimi není podstatný vizuální rozdíl, tudíž při pouhém zpracování obrazu nejdou vzájemně odlišit. Výsledek segmentace je zobrazen na obrázku 4.7a.

Segmentace troilitu

Troilit je nerost patřící do sulfidů a v meteoritu se vyskytuje v podobě šedivých shluků. Nicméně jeho detekce na základě pouze šedivé barvy není možná, protože šedivé oblasti, nacházejí se i v bezprostřední blízkosti kovů, nejsou troilitem. Proto



(a) Segmentované části železa nebo niklu. (b) Segmentované části troilitu.

Obr. 4.7: Ukázka segmentace částí meteoritu.

byla pro segmentaci zvolena metoda trénovatelné segmentace, která při klasifikaci jednotlivých pixelů uvažuje kromě barvy i hodnoty okolí pixelu.

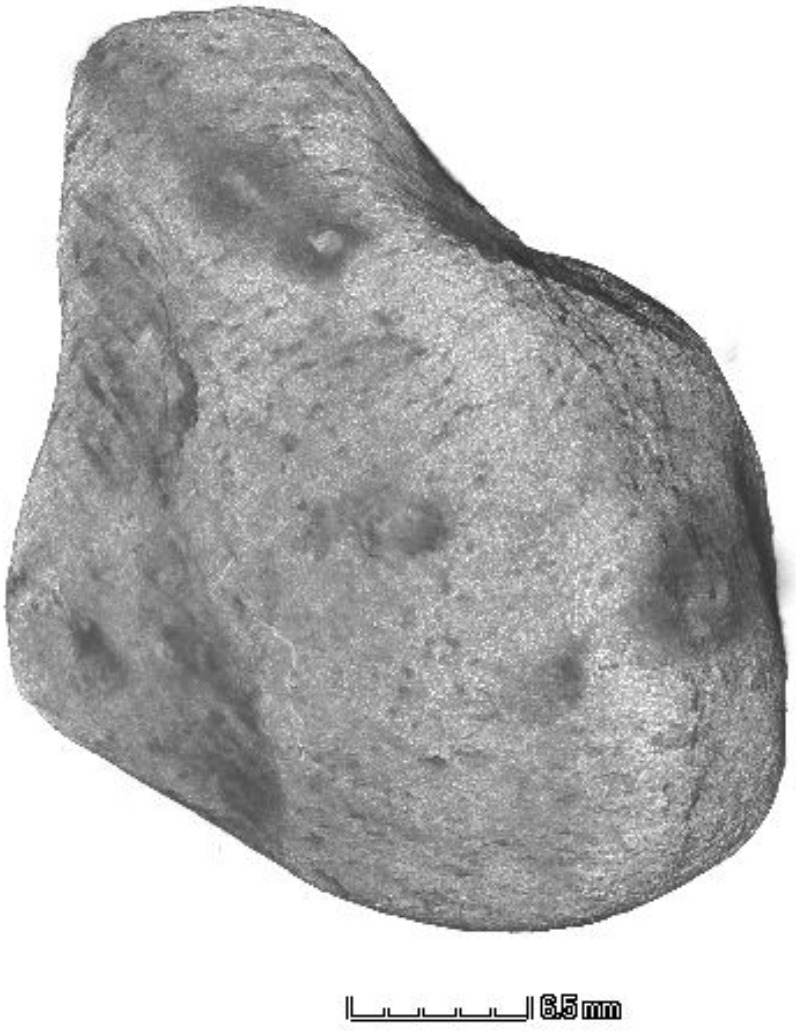
Parametry trénovatelné segmentace byly nastaveny stejně jako v případě segmentace meteoritu (uvedeno výše). Klasifikační model byl natrénován na jednom referenčním snímku (viz obrázek 4.6a) a aplikován na snímky všech tří meteoritů. Výsledek trénovatelné segmentace je zobrazen na obrázku 4.7b.

4.2.3 Dosažené výsledky

Výsledkem celého procesu bylo spočítat objemové zastoupení jednotlivých materiálů obsažených v meteoritu. To bylo provedeno na základě spočtených velikostí segmentovaných částí každého 2D obrazu a díky známé velikosti voxelu (pixel v prostoru) byly spočteny objemy pro celý 3D obraz. V tabulce 4.4 jsou zobrazeny vypočtené objemy meteoritů a materiálů v nich obsažených. Z toho 2,4 % všech tří meteoritů je tvořeno kovy a 2,9 % je zastoupen troilit. Vizualizace 3D obrazu druhého meteoritu byla provedena nástroji z pracoviště CEITEC a výsledek je zobrazen na obrázku 4.8. Výsledkem tohoto výzkumu provedeného společně s CEITEC byl závěr, že všechnu tři meteority patřily zřejmě do stejného roje meteoritů, i když byly nalezeny v různém časovém období.

Tab. 4.4: Vypočtené zastoupení jednotlivých segmentovaných částí.

	Objem met. [mm^3]	Objem Fe-Ni [mm^3]	Zastoupení Fe-Ni [%]	Objem troilitu [mm^3]	Zastoupení troilitu [%]
Met. 1	1801	55	3	63	3.5
Met. 2	12985	348	2,7	311	2,4
Met. 3	12887	261	2	429	3,3



Obr. 4.8: Vizualizace 3D dat meteoritu.

4.3 Testování optimalizované verze algoritmu *k*–nejbližších sousedů

V této kapitole budou popsány dosažené výsledky při testování algoritmu k–nejbližších sousedů na větších trénovacích souborech dat. Cílem testování nebylo změřit přesnost natrénovaného modelu, ale ověřit a změřit zrychlení optimalizované verze algoritmu.

Kvůli zvětšujícím se objemům dat vzniká potřeba tato rozsáhlá data zpracovávat nejlépe v reálném čase. Problémem je, že současné volně dostupné programy (jakými jsou RapidMiner[1], Weka[2], Knime[3], ...) nejsou navrženy tak, aby dostatečně rychle dokázaly zpracovávat velké objemy dat. Na základě provedených experimentálních testů s těmito programy, které vyloučily použití stávajících algoritmů pro zpracování velkých objemů dat, byl jeden z nejznámějších učících se algoritmů umělé inteligence upraven pro nasazení na grafických akcelerátorech. Optimalizovaný algoritmus k–nejbližších sousedů, vytvořený v rámci této práce v prostředí CUDA a OpenCL byl již publikován v [3, 4].

4.3.1 Použitá data

Pro testování bylo použito několik různě velkých databází s různým počtem atributů i celkových prvků. Tyto databáze byly uměle vygenerovány s pomocí prostředí programu RapidMiner. Data byla vygenerována s polynomiálním rozložením a to vždy tak, že trénovací soubor dat obsahoval 25 % prvků a testovací soubor dat obsahoval 75 % prvků. Soubory dat byly vytvořeny pro rozsáhlejší testování algoritmů a čítaly od 0,4 milionu po 4 miliony prvků. Zvolené počty atributů souborů dat byly od 4 do 1000 atributů.

4.3.2 Dosažené výsledky

Testy byly provedeny s použitím jednoho, dvou, třech a čtyřech grafických akcelerátorů. Testoval se jak algoritmus vytvořený v prostředí CUDA, tak i v prostředí OpenCL. Pro srovnání s akcelerovanými verzemi algoritmu byla použita CPU verze k–NN algoritmu z prostředí programu RapidMiner. Výsledky klasifikace byly pro všechny verze algoritmů stejné. Rychlost výpočtu jednotlivých algoritmů bude popsána níže.

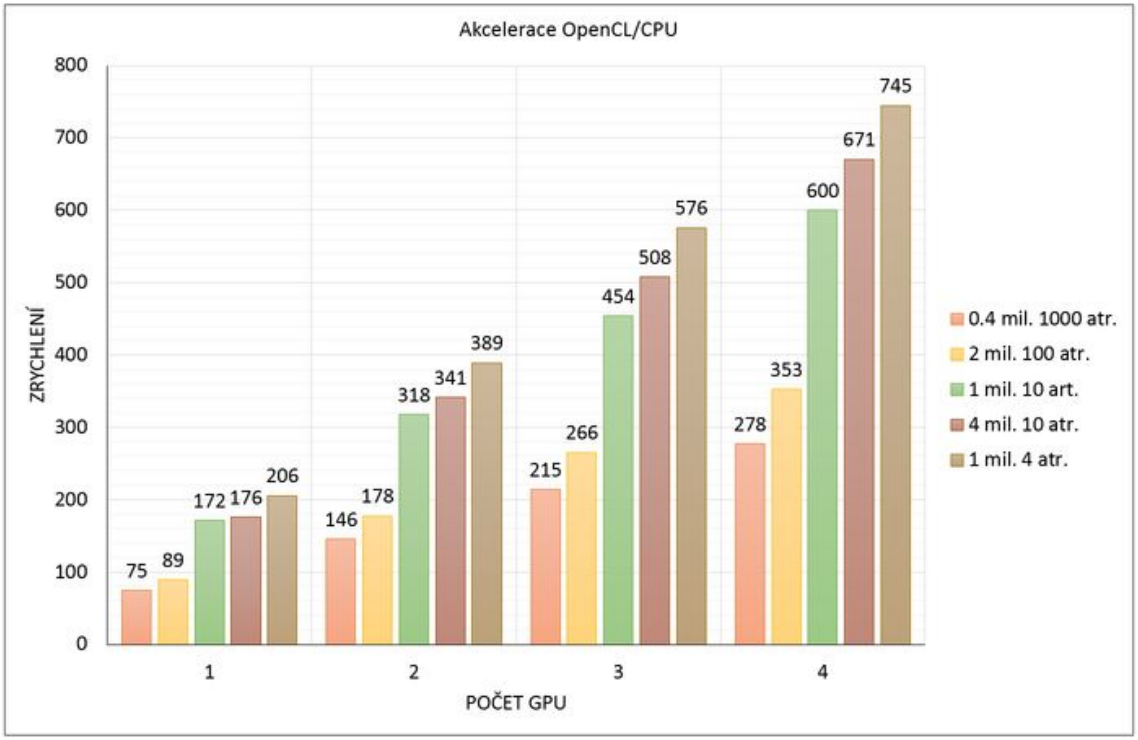
[1] Dostupné z URL: `http://rapidminer.com/`
[2] Dostupné z URL: `http://www.cs.waikato.ac.nz/ml/weka/`
[3] Dostupné z URL: `http://www.knime.org/`

Tabulka 4.5 popisuje výsledky dosažené pro OpenCL implementaci algoritmu a v tabulce 4.6 jsou popsány výsledky pro CUDA implementaci. Pro tyto testy byl nastaven parametr $k = 5$, protože je to běžně uživateli nastavovaná hodnota.

Tab. 4.5: Výsledky výpočtů pro OpenCL implementaci $k = 5$.

	1 CPU	1 GPU	2 GPUs	3 GPUs	4 GPUs
0,4 mil., 1000 atributů	17h 35min	843 s	434 s	295 s	228 s
2 mil., 100 atributů	2d 4h 10min	2106 s	1056 s	707 s	532 s
1 mil., 10 atributů	1h 55min	40,2 s	21,7 s	15,2 s	11,5 s
4 mil., 10 atributů	1d 7h 29min	645 s	332 s	223 s	169 s
1 mil., 4 atributy	1h 12min	21 s	11,1 s	7,5 s	5,8 s

CPU - Intel Core i7 3770@4.1GHz, L3 paměť - 8192kB
4 GPU - 2x3072 jader, paměť. 2x4096MB@6 GHz, GPU - 1019 Mhz



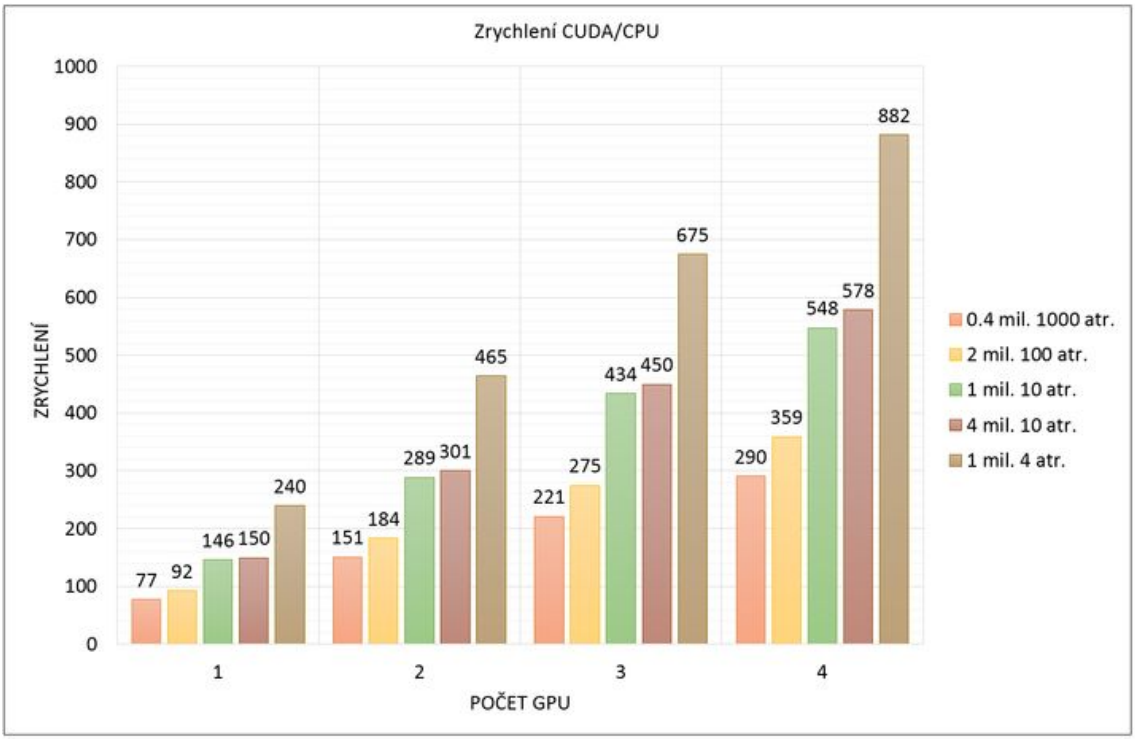
Obr. 4.9: Akcelerace OpenCL verze algoritmu k-NN.

Z výsledků je zřejmé, že výpočetní úloha trvající na CPU dny, může být při nasazení na GPU spočtena během několika minut. Obrázek 4.9 zobrazuje celkové zrychlení vytvořené OpenCL verze algoritmu. Z grafu dále vyplývá, že při zvyšujícím se počtu atributů klesá zrychlení algoritmu. Naopak se zvyšující se velikostí databáze se zrychlení zvyšuje. Výsledky zrychlení mezi CPU verzí algoritmu a verzí algoritmu

Tab. 4.6: Výsledky výpočtů pro CUDA implementaci $k = 5$.

	1 CPU	1 GPU	2 GPUs	3 GPUs	4 GPUs
0,4 mil., 1000 atributů	17h 35min	818 s	419 s	286 s	218 s
2 mil., 100 atributů	2d 4h 10min	2038 s	1023 s	683 s	523 s
1 mil., 10 atributů	1h 55min	47,3 s	23,9 s	15,9 s	12,6 s
4 mil., 10 atributů	1d 7h 29min	757 s	377 s	252 s	196 s
1 mil., 4 atributy	1h 12min	18 s	9,3 s	6,4 s	4,9 s

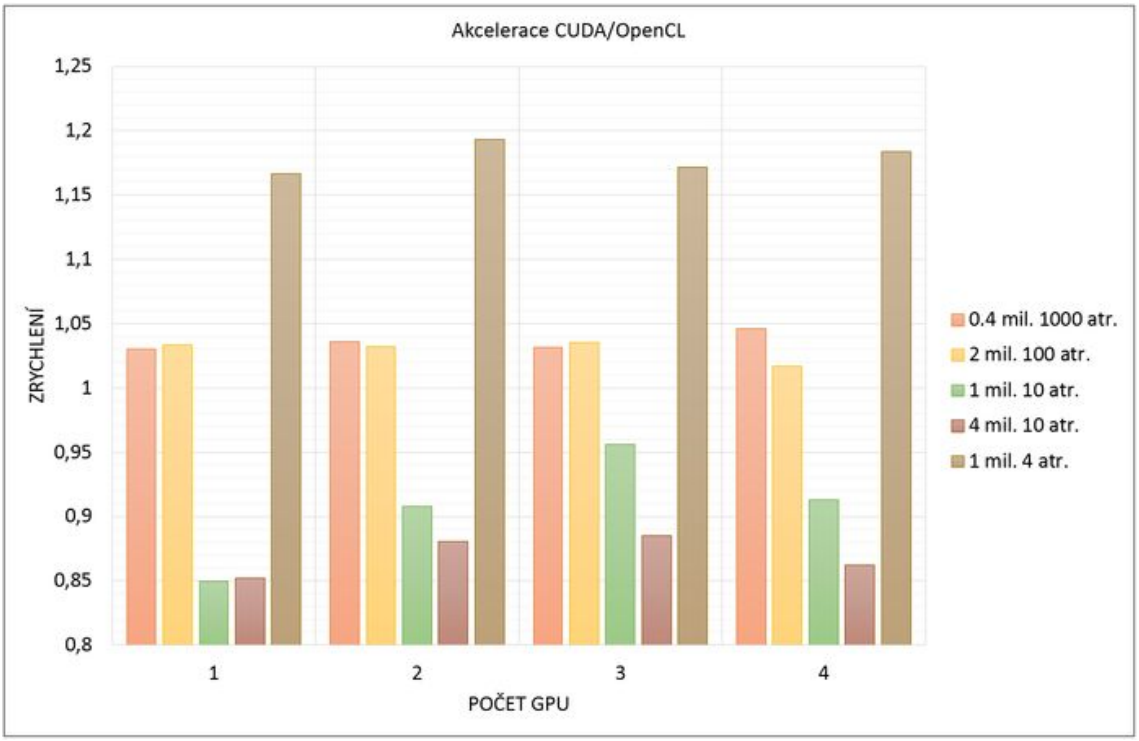
CPU - Intel Core i7 3770@4.1GHz, L3 paměť - 8192kB
4 GPU - 2x3072 jader, paměť. 2x4096MB@6 GHz, GPU - 1019 Mhz



Obr. 4.10: Akcelerace CUDA verze algoritmu k-NN.

vytvořenou v prostředí CUDA, jsou zobrazeny na obrázku 4.10. Nejvyšší zrychlení bylo dosaženo pro testovací soubor dat obsahující čtyři atributy a čítající milion prvků. Toto výsledné zrychlení bylo 882-ti násobné.

Dále byly vzájemně mezi sebou srovnány implementace obou verzí GPU algoritmů. Z výsledků na obrázku 4.11 je vidět, že pro soubory dat s počty atributů 100 a 1000 byla varianta algoritmu vytvořená v prostředí CUDA o 3 % rychlejší než varianta pro prostředí OpenCL. Pro soubory dat s 10-ti atributy bylo OpenCL varianta o 10 % rychlejší a pro 4 atributy o 15 % pomalejší než varianta vytvořená v prostředí CUDA. Tyto rozdíly pro zrychlení jednotlivých verzí GPU algoritmu pro



Obr. 4.11: Porovnání zrychlení mezi implementacemi CUDA a OpenCL algoritmu k-NN.

soubory dat se 4-mi a 10-ti atributy jsou pravděpodobně způsobeny schopností každého prostředí pracovat s jinou efektivitou s použitým datovým typem *float4*, kde pokud je zvolen soubor dat se 4-mi atributy (hodnoty obsaženy v jednom datovém poli typu *float4*), dokáže prostředí CUDA pracovat efektivněji, než když je zvolen soubor dat s 10-ti atributy, kde je nutnost použít 3 datová pole typu *float4*, přičemž 2 hodnoty alokovaného pole zůstanou pro každý prvek nevyužity. Při výpočtu průměrného zrychlení pro všechna měření bylo prostředí CUDA o 0,5 % rychlejší, než prostředí OpenCL.

Tab. 4.7: OpenCL - srovnání pro různá k.

	$k = 1$	$k = 10$	$k = 20$
1 mil., 10 atributů	14,1 s	25,2 s	93,7 s
1 mil., 4 atributy	4,1 s	17,8 s	88 s

Kvůli výrazné modifikaci původního k-NN algoritmu byly provedeny i další testy pro vyšší hodnoty parametru ($k = 10$ a $k = 20$). Tabulky 4.7 a 4.8 zobrazují výsledky pro různá k (testováno pro 4 GPU). Z výsledků vyplývá, že algoritmy dosahují požadovaného zrychlení pro $k < 10$. Při vyšším počtu k nejbližších sousedů je zrych-

lení algoritmů velmi sníženo. Z výsledků dále vyplývá, že pro algoritmus k-NN je prostředí CUDA mírně rychlejší, než prostředí OpenCL.

Tab. 4.8: CUDA - srovnání pro různá k.

	$k = 1$	$k = 10$	$k = 20$
1 mil., 10 atributů	16 s	21,7 s	88,5 s
1 mil., 4 atributy	3,8 s	16,2 s	78,8 s

Úspora elektrické energie

Úspora elektrické energie je spočítána pro scénář s použitím algoritmu optimalizovaného v prostředí CUDA a databází čítající 1 milión prvků a čtyři atributy. Výpočetní čas v případě použití 4 GPU byl 4,9 s, což při příkonu 600 W odpovídá spotřebě elektrické energie $600 \cdot \frac{4,9}{3600} = 0,81$ Wh. V případě použití jednoho CPU byl výpočetní čas pro stejný scénář 1h 12min. Při použití všech 4 jader CPU je odpovídající zrychlení procesu čtyřnásobné. To znamená, že celkový čas výpočtu byl 1080 s a při uvažovaném příkonu 180 W byla spotřeba elektrické energie $180 \cdot \frac{1080}{3600} = 54$ Wh. Při tomto scénáři bylo dosaženo asi 66-ti násobného ušetření elektrické energie při použití algoritmů běžících na grafických akcelerátorech.

4.3.3 Srovnání s jinými pracemi

Vytvořený algoritmus k-NN byl srovnán s jinými současnými řešeními. Pro testy byl použit běžný grafický akcelerátor Nvidia sloužící primárně v zábavním průmyslu pro hraní 3D her. Oproti tomu ve většině jiných prací je pro výpočet použit profesionální hardware Nvidia Tesla. Vytvořený algoritmus může být nasazen na systémy s více grafickými akcelerátory pro dosažení většího zrychlení. Verze algoritmů publikovaných v jiných pracech jsou vytvořeny obvykle pro nasazení na jednom GPU.

Tab. 4.9: Porovnání řešení pro GPU optimalizaci algoritmu k-NN.

Řešení	Zrychlení [-]	Typ GPU	Více GPU
Liang a kol. (2009) [63]	47x	Nvidia Tesla C1060	ne
Komarov a ko. (2014) [96]	80x	Nvidia Tesla C2050	ne
Gavahi a kol. (2015) [97]	110x	Nvidia Tesla K20x	ne
Tang a kol. (2015) [98]	336x	Nvidia Tesla C2075	ne
Navržená metoda	**882x**	Nvidia GTX 690	ano

V [63] použili autoři různé optimalizační techniky pro dosažení 47-mi násobného zrychlení oproti CPU verzi algoritmu. Práce [96] se zabývala optimalizováním použité metody rychlého výběru (Quick Select) při dosažené 80-ti násobné akceleraci. V [97] byly matematickými operacemi eliminovány opakující se výpočty, kde výsledkem bylo 110-ti násobné zrychlení. Práce [97] se zabývala optimalizací různých řadících technik. Dosažená akcelerace byla 336-ti násobná. V tabulce 4.9 jsou porovnána jednotlivá řešení. Nejvyššího zrychlení dosáhla nově vytvořená metoda, kde bylo dosaženo 882-ti násobného zrychlení, kdy byl algoritmus spuštěn na 4 GPU akcelerátorech.

4.4 Trénování optimalizovaného obličejového detektoru

V této části budou popsány výsledky trénování a testování objektového detektoru, který byl optimalizován pro grafické akcelerátory. Pro otestování optimalizovaného algoritmu byl zvolen problém detekce obličejů. Tato úloha vyžaduje použití větších trénovacích databází pro získání dostatečné klasifikační přesnosti. Tímto požadavkem je výrazně zvýšena doba trénování detektoru a proto nebylo cílem změřit přesnost detekce natrénovaného modelu, ale změřit zrychlení optimalizované verze algoritmu oproti CPU verzi algoritmu. Funkčnost natrénovaného detektoru byla zběžně ověřena na několika snímcích. Testovací proces natrénovaného detektoru pro GPU optimalizován nebyl z důvodů velmi rychlé detekce většiny řešených problémů při nasazení na více-jádrovém CPU. Výsledky byly již publikovány v [37].

4.4.1 Použitá data

Pro trénování a testování byly použity snímky z volně dostupné obličejové databáze MIT CBCL [99] a pro testování z obličejové databáze CMU [100]. Trénování probíhalo na 2429 snímcích obličejů (viz obrázek 4.12a) a 4548 snímcích pozadí (viz obrázek 4.12b). Tyto snímky byly zvětšeny z původních rozměrů 19x19 pixelů na rozměry 24x24 pixelů, protože původní nastavení velikosti trénovacího okna podle metody Viola-Jones pro detekci obličejů bylo právě 24x24 pixelů. Pro trénování bylo tedy použito 6977 snímků a pro otestování funkčnosti natrénovaného detektoru bylo vybráno pouze několik snímků.

(a) Positivní snímky (b) Negativní snímky

Obr. 4.12: Detekce obličejů - ukázka trénovacích dat.

4.4.2 Dosažené výsledky

Pro výpočty byl použit počítač s procesorem Intel Core i7-3770 s frekvencí procesoru 3,7 GHz, 16 GB pamětmi a s dvěma dvou jádrovými GPU GeForce GTX 690. Celkový výpočetní výkon systému byl 11,24 TFLOPS. Trénování detektoru probíhalo na CPU s jedním použitým jádrem, dále postupně na 1 až 4 grafických akcelerátorech. Vstupem trénovacího procesu bylo 6977 snímků s rozlišením 24x24 pixelů a dále 189 664 vygenerovaných Haarových příznaků.

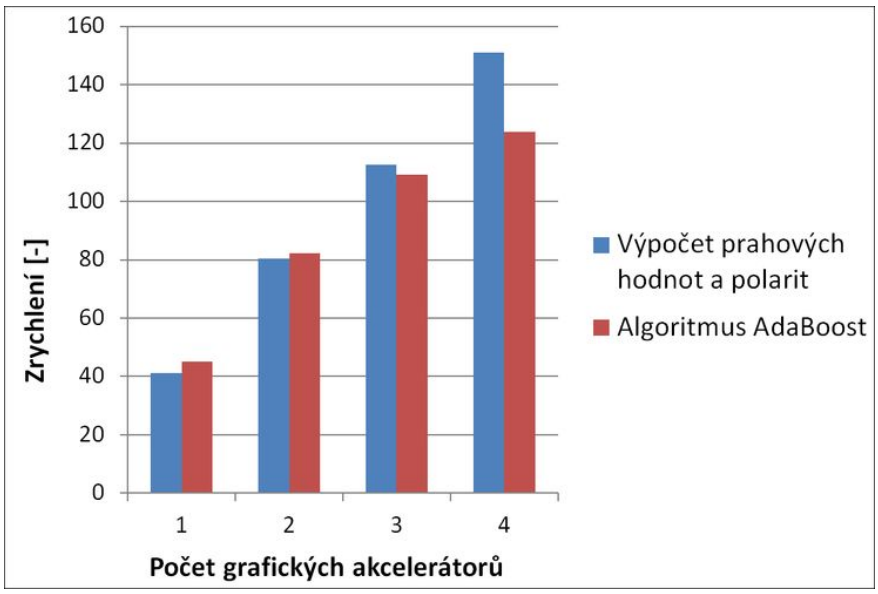
Tab. 4.10: Výsledky výpočtů - trénovaní obličejového detektoru.

	1 CPU	1 GPU	2 GPU	3 GPU	4 GPU
Prahové hodnoty & polarity	20 250 s	492 s	252 s	180 s	134 s
Algoritmus AdaBoost	136,3 s	3,02 s	1,66 s	1,25 s	1,1 s

CPU - Intel Core i7 3770@3.7Ghz, L3 cache - 8192kB
4 GPU - 2x3072 jader, paměti 2x4096MB@6 GHz, GPU - 1019 Mhz

Tabulka 4.10 zobrazuje dobu trénování detektoru pro jednotlivé optimalizované funkce, kde první funkce hledá optimální parametry všech lineárních klasifikátorů a druhá funkce hledá nejlepší klasifikátory s pomocí funkce AdaBoost, kde zobrazený čas trvání je pro nalezení právě jednoho lineárního klasifikátoru. Při použití CPU byl čas výpočtu pro použití první optimalizované funkce 5 hodin a 40 minut ve srovnání s 134 s, pokud běžel výpočet na 4 GPU. Dále pro druhou optimalizovanou funkci byl nejlepší klasifikátor vyhledán s pomocí CPU za 136,6 s v porovnání s 1,1 s při použití 4 GPU. Nejvyšší zrychlení (viz obrázek 4.13) bylo tedy pro první funkci 151 násobné a pro druhou funkci 124 násobné při použití 4 GPU. Nižší zrychlení pro druhou optimalizovanou funkci je dosaženo z důvodů rychlého výpočtu jedné iterace

(nalezení nejlepšího klasifikátoru) a tím opakovaného kopírování dat do GPU a zpět, což je časově náročné. Výsledný natrénovaný detektor se skládal z 3000 klasifikátorů a čas trénování na CPU byl přibližně 5 dnů. Při použití 4 GPU trval výpočetní proces přibližně jednu hodinu.



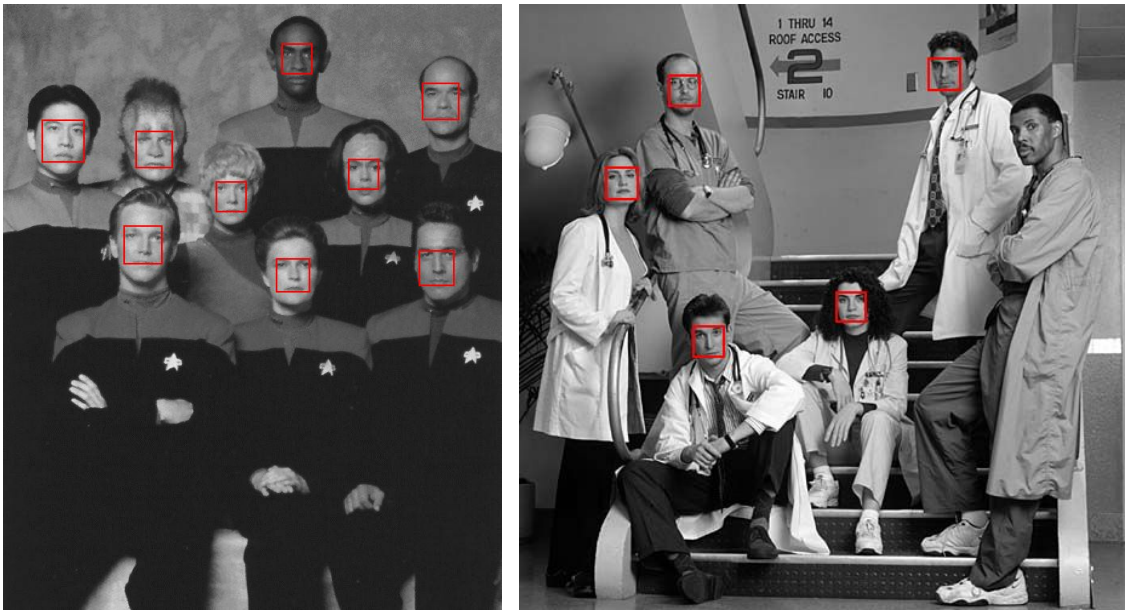
Obr. 4.13: Zrychlení funkcí optimalizovaných pro GPU.

4.4.3 Výsledky detekce

Natrénovaný detektor byl poté aplikován na několik vybraných snímků, kde výsledky detekce jsou zobrazeny na obrázku 4.14a, kde byly správně detekovány všechny obličeje a na obrázku 4.14b, kde nebyl detekován jeden obličej z důvodu natočení hlavy člověka a zobrazení pouze profilu obličeje. Detektor je natrénován pouze na obličejích z předního pohledu. Testování přesnosti detekce nebylo detailně zkoumáno z důvodů, že Viola-Jones detektor je všeobecně znám pro řešení problému trénování obličejového detektoru a výsledky přesnosti tohoto detektoru, které jsou dostačující, byly již mnohokrát publikovány [20, 91].

Úspora elektrické energie

Úspora energie byla spočtena pro optimalizovaný algoritmus nastavující parametry klasifikátorů. V případě použití čtyř GPU trval výpočet 134 s. Výsledná spotřeba elektrické energie byla $600 \cdot \frac{134}{3600} = 22\,\text{Wh}$. Výpočetní čas pro 1 CPU byl 20 250 s. Dosažená spotřeba činila $180 \cdot \frac{1}{4} \cdot \frac{20\,250}{3600} = 253\,\text{Wh}$. To znamená, že při použití verze GPU algoritmu na grafických akcelerátorech, je spotřebováno 11,5x méně elektrické energie.



(a) Všechny obličeje detekovány. (b) Jeden nedetekovaný obličej.

Obr. 4.14: Experimentální výsledky detekce obličejů (snímky převzaty z [99]).

4.4.4 Srovnání s jinými pracemi

Optimalizováním detekčního procesu s pomocí GPU se v dnešní době zabývá mnoho prací [30, 31, 32]. Protože pro problémy řešené v rámci této práce byla dostačující optimalizovaná CPU verze detekčního algoritmu, nebylo třeba provádět optimalizaci pro GPU. Oproti tomu trénování kaskádního modelu s pomocí algoritmu AdaBoost se zabývá pouze minimální počet prací. Zde bude porovnána vytvořená metoda s optimalizovanou verzí algoritmu AdaBoost autorů práce [101].

Tab. 4.11: Porovnání metod pro GPU optimalizaci algoritmu pro trénování detektoru.

Řešení	Zrychlení [-]	Typ GPU	Počet GPU
Tsai a kol. (2015) [101]	34x	Nvidia Tesla K20c	1
Nová metoda	**137x**	Nvidia GTX 690	4

Tabulka 4.11 zobrazuje srovnání metod. Vytvořená nová metoda dosáhla v průměru 137-mi násobného zrychlení oproti CPU verzi algoritmu. Vyšší zrychlení bylo v porovnání s prací [101] dosaženo zejména díky optimalizaci pro spouštění na více GPU.

4.5 Detekce artérie z ultrazvukových snímků

V rámci této části práce byla řešena rychlá a velmi přesná lokalizace artérie v ultrazvukových snímcích zdravých pacientů. V prvotní verzi, která byla publikována v impaktovaném článku [1] (IF=2,1), byla dosažena přesnost detekce 97 % na databázích zdravých pacientů. Při otestování tohoto natrénovaného detektoru na nemocných pacientech (arterie obsahovala různé množství plaku uvnitř její stěny) byla přesnost detekce 89 % pro pacienty s menší mírou plaku v tepnách a 44 % pro pacienty s vyšší mírou plaku. V rámci této práce byl původní publikovaný postup rozšířen. Byla vytvořena nová trénovací databáze zahrnující i data nemocných pacientů. Dále byl na těchto databázích natrénován a evolučně optimalizován detektor, jehož přesnost byla na datech nemocných pacientů 95 % a na datech zdravých pacientů 96 %. S touto kapitolou souvisí kapitola 4.6, která se zabývá trénováním a testováním modelu hlubokého učení (konvoluční neuronové sítě), kde je tento model použit pro klasifikaci detekované tepny do 3 kategorií (zdravá arterie, nižší úroveň plaku arterie, vyšší úroveň plaku arterie).

4.5.1 Použité databáze snímků arterie

Ultrazvuková snímky použité pro trénování a testování detektoru byly nasnímány s pomocí tří ultrazvukových přístrojů. Data byla nasnímána na celkem 21 pacientech, kde 75 video sekvencí patřilo zdravým a 4 video sekvence nemocným pacientům. Databáze obsahuje data 18 zdravých pacientů ve věku 25 až 40 let, jednoho třináctiletého zdravého dětského pacienta a 2 nemocných pacientů ve věku přes 70 let. Všichni pacienti souhlasili s poskytnutím jejich dat pro výzkumné účely.

Ultrazvukové přístroje

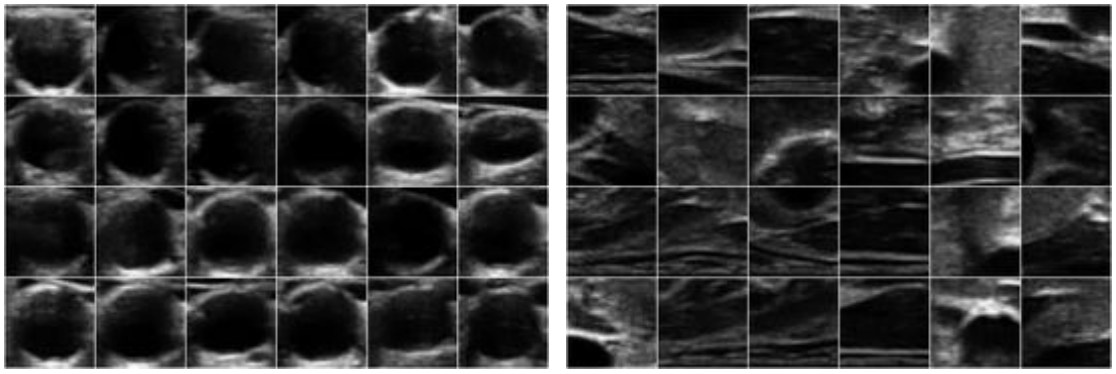
Pro získání použitých video sekvencí byly, jak již bylo řečeno, použity 3 různé ultrazvukové přístroje:

1. Ultrasonix OP s lineární sondou L14–5/38 (výrobce - Ultrasonix Medical, Richmond, BC, Kanada). Rozlišení snímků 388x400 pixelů.
2. Toshiba Nemio XG s lineární sondou o frekvenci 7.5 MHz (výrobce - Toshiba Medical Systems, Shimoishigami, Otawara-Shi, Tochigi-Ken, Japonsko). Rozlišení snímků 226x322 pixelů.
3. Philips s lineární sondou L11-3 s rozsahem frekvencí od 11 Hz do 3 MHz. Rozlišení snímků 389x252 pixelů.

Z těchto získaných video sekvencí byly následně vybrány snímky a vytvořeny trénovací a testovací databáze.

Trénovací snímky

Pro vytvoření trénovací databáze byla použita jak data zdravých pacientů z ultrazvukového přístroje Ultrasonix, tak i data nemocných pacientů z přístroje Philips. Původní trénovací databáze z přístroje Ultrasonix byla již použita v práci [1] a bude použita pro srovnání výsledků při použití detektorů natrénovaných i na nemocných pacientech. Tato databáze obsahuje 283 snímků, které byly získány ze tří video sekvencí jednoho pacienta.

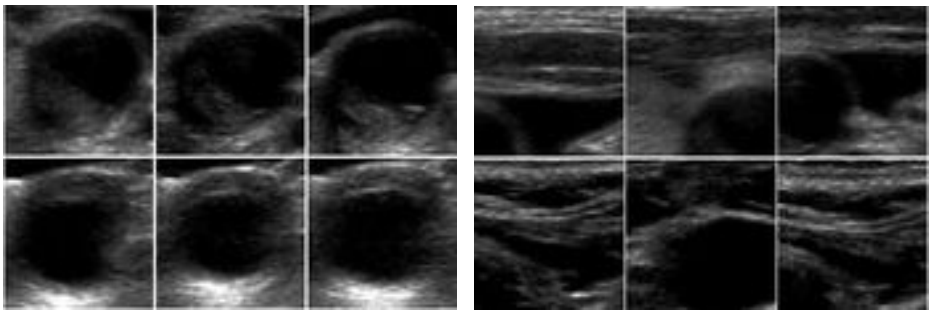


(a) Positivní snímky (b) Negativní snímky

Obr. 4.15: Detekce příčného řezu arterie - ukázka původních trénovacích dat.

Druhá trénovací databáze byla získána z ultrazvukového přístroje Philips a obsahuje 207 snímků nemocných pacientů, kde 102 snímků pacienta s nižší úrovní plaku bylo získáno z jedné video sekvence a 105 snímků pacienta s vyšší úrovní plaku bylo rovnoměrně získáno ze 3 video sekvencí.

Pro vytvoření trénovacích positivních (obsahují hledanou artérii) a negativních snímků (obsahují pozadí snímku) o rozměrech 24x24 pixelů bylo vybráno 200 snímků zdravých (viz obrázek 4.15) pacientů z databáze Ultrasonix a 207 snímků nemocných (viz obrázek 4.16) pacientů z databáze Philips. Databáze pro trénování kaskádních modelů se tedy skládala z 407 positivních a 407 negativních snímků. Získané trénovací databáze obsahovaly navíc hodně zašumělé snímky artérie nebo mírně zdeformované snímky vlivem přítlaku sondy v blízkosti analyzované oblasti.



(a) Positivní snímky (b) Negativní snímky

Obr. 4.16: Detekce příčného řezu arterie - ukázka dat nemocných pacientů.

Testovací snímky

Testování probíhalo na databázích získaných z video sekvencí všech tří ultrazvukových přístrojů. Testovací databáze z přístrojů Ultrasonix a Toshiba byly již uvedeny v impaktovaném článku [1] autora této práce a zde budou použity pro porovnání přesností nově natrénovaných detektorů s již publikovanými výsledky.

Testovací databáze z ultrazvukového přístroje Ultrasonix čítá 538 snímků, které byly rovnoměrně získány z 50-ti video sekvencí 15-ti dospělých pacientů. Z přístroje Toshiba bylo rovnoměrně získáno 433 snímků z celkem 22 video sekvencí od jednoho dětského (menší rozměr arterie) a dvou dospělých pacientů. Poslední databáze snímků čítající 486 snímků byla získána z video sekvencí obou nemocných pacientů, pořízených na přístroji Philips. Od pacienta s nižším výskytem plaku v artérii bylo použito 385 snímků a od pacienta s vyšším výskytem plaku bylo použito 101 snímků. Tyto snímky byly sice získány ze stejných video sekvencí jako trénovací databáze, ale byly získány v jiných časových intervalech než snímky trénovací.

Vzhledem k tomu, že některé video sekvence byly zaznamenány při dýchání nebo polykání, je testování více vypovídající díky dosažené různorodosti snímků.

4.5.2 Dosažené výsledky

Tato část se zabývá popisem trénování a následného testování různých kaskádních modelů pro rychlou a přesnou detekci artérie v příčném řezu ultrazvukového snímku. Cílem trénování bylo vytvořit robustní detektor, který by dosahoval dostačujících výsledků jak pro zdravé, tak i pro nemocné pacienty. K tomu byla vytvořena trénovací databáze skládající se z dat zdravých i nemocných pacientů. V dalším textu budou zhodnoceny použité trénovací a testovací metody a popsány nastavené parametry.

Parametry trénování

V rámci práce byly primárně natrénovány tři kaskádní modely založené na příznacích Haar, HOG a LBP. Dále byla na základě výsledků jako nejúspěšnější zvolena metoda Haar. Dodatečně byla ještě natrénována kaskáda modelů umělé inteligence (s využitím k-NN a metody Haar). Parametry trénovacího procesu byly nastaveny takto: velikost trénovacího okna 24x24 pixelů, počet nejlepší nalezených klasifikátorů byl 350. Natrénovaný kaskádní model s nejvyšší přesností a rychlostí byl dále optimalizován s pomocí evolučních algoritmů pro dosažení ještě přesnějších výsledků. Podrobné výsledky jednotlivých kaskádních modelů budou popsány dále.

Parametry detekce

Detekční proces měl nastaveny následující parametry: startovní velikost detekčního okna byla 72x72 pixelů, hodnota zvětšení detekčního okna během každé iterace byla 1,1, hodnota parametru posunu detekčního okna byla v každé iteraci 0,1. Startovní velikost detekčního okna byla za účelem snížení výpočetní náročnosti zvolena podle očekávané minimální velikosti arterie dospělého člověka.

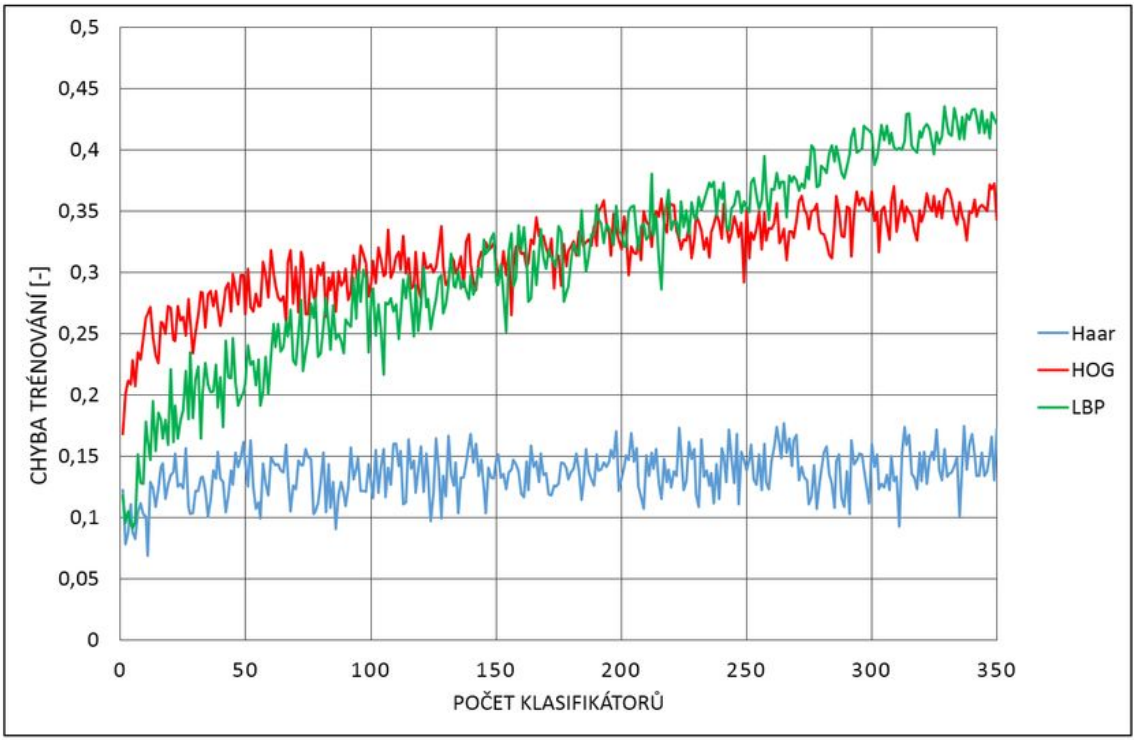
Z důvodů, že analyzované snímky mohou také obsahovat snímky dětských pacientů s menším rozměrem arterie, než je startovní velikost detekčního okna, byl detekční algoritmus modifikován. V případě, že detekční algoritmus nenalezne žádnou arterii, je startovní velikost detekčního okna snížena na 48x48 pixelů a proces probíhá znovu.

Při zpracování detekovaných oblastí v rámci každého snímku byly uvažovány dva scénáře: se zobrazením všech detekcí a se zobrazením jen té nejvýznamnější detekce.

Výsledky trénování a testování

Nejprve byly na základě vstupních 407 positivních a 407 negativních snímků natrénovány kaskádní modely založené na metodách Haar, HOG a LBP, kde trénovací algoritmus AdaBoost vybíral 350 nejlepších klasifikátorů (počet klasifikátorů byl zvolen experimentálně). Na obrázku 4.17 je zobrazena chyba trénování během hledání nejlepších klasifikátorů. Z těchto nejlepších klasifikátorů byly dále poskládány a vytvořeny kaskádní modely podle stejně definovaného schématu, kde první stupeň kaskády obsahoval 3 klasifikátory, druhý stupeň 6, třetí stupeň 25 a následující stupně vždy po padesáti klasifikátorech. Tímto navrženým schématem bylo dosaženo rychlé detekční funkce jednotlivých kaskádních modelů.

Dále je z obrázku 4.17 patrné, že trénovací chyba pro metodu Haar je ze všech nejnižší. Ostatním metodám trénovací chyba roste výrazně rychleji. Toto je dáno



Obr. 4.17: Trénovací chyba jednotlivých typů zvolených příznaků.

řešeným problémem, kde každá metoda je vhodná na jiný typ dat a dále také celkovým množstvím příznaků, ze kterých je možné vybírat (Haar - 189 664, HOG - 10 000, LBP 1 458), protože například u metody LBP jsou dřívěji vybrány vhodné příznaky a zbývají ty již méně vhodné.

Takto natrénované modely byly testovány dvěma způsoby: s uvažováním všech možných detekcí a s uvažování pouze jedné nejvýznamnější detekce (za předpokladu, že každý ultrazvukový snímek obsahuje právě jednu artérii). Při uvažování všech možných detekcí byla počítána přesnost detekce (míra správných detekcí) $R_{\text{TP}} = \frac{N_{\text{TP}}}{N_{\text{I}}}$ a míra falešně positivních detekcí $R_{\text{FP}} = \frac{N_{\text{FP}}}{N_{\text{I}}}$, kde N_{TP} označuje počet správně detekovaných objektů, N_{FP} označuje celkový počet falešně positivních detekcí a N_{I} označuje celkový počet testovacích snímků. V rámci jednoho snímků může být detekováno více falešně positivních detekcí, proto je možné dosáhnout hodnoty R_{FP} větší než 100 %. Při uvažování pouze jedné nejvýznamnější detekce je spočítána pouze přesnost detekce R_{TP}.

Testování natrénovaných kaskádních modelů probíhalo na 5-ti databázích ultrazvukových snímků, které byly vytvořeny na základě typu ultrazvukového přístroje a dále rozděleny na trénovací a testovací část. Z tabulek 4.12 (pro zdravé pacienty) a 4.13 (pro nemocné pacienty) jsou patrné výsledky přesnosti detekce a míra falešně positivních detekcí. Z výsledků pro základní 4 kaskádní modely (mimo Haar & evol.) vyplývá, že metoda Haar celkově dosáhla jako jediná vysokých přesností detekce a

zároveň nízké míry falešně positivních detekcí. Například metoda LBP dosahovala vyšších přesností než metoda Haar, ale na úkor míry falešně positivních detekcí, která se pohybovala ve stovkách procent.

Tab. 4.12: Přesnost detekce pro databáze zdravých pacientů (všechny výskyty arterie v obraze).

	Trénovací data Ultrasonix (283)		Testovací data Ultrasonix (538)		Testovací data Toshiba (433)	
Kask. model	R_{TP} [%]	R_{FP} [%]	R_{TP} [%]	R_{FP} [%]	R_{TP} [%]	R_{FP} [%]
Haar	91,51	24,1	98,14	24,9	82,67	6,4
HOG	88,33	38,9	97,39	50,7	16,39	5,1
LBP	99,29	1121	99,25	921	93,53	326
Haar & k-NN	83,39	28,2	98,88	30,8	59,58	5,3
Haar & evol.	95,4	29,3	97,76	29,3	87,6	9

Tab. 4.13: Přesnost detekce pro databáze nemocných pacientů (všechny výskyty arterie v obraze).

	Trénovací data Philips (207)		Testovací data Philips (486)	
Kask. model	R_{TP} [%]	R_{FP} [%]	R_{TP} [%]	R_{FP} [%]
Haar	97,1	24,1	87,57	14,8
HOG	22,22	26,5	24,48	12
LBP	99,51	615	97,55	640
Haar & k-NN	84,54	55,1	86,15	22,2
Haar & evol.	96,13	61,3	92,05	29,4

Více vypovídající pro řešení problému detekce arterií jsou tabulky 4.14 a 4.15, kde byl uvažován pouze výskyt jedné arterie v obrazu a proto spočtena pouze úspěšnost detekce. Při srovnání prvních čtyřech kaskádních modelů dosáhla nejvyrovnanějších výsledků přesnosti metoda Haar. Metoda HOG dosáhla velmi nestabilních výsledků, kdy na jednu stranu dosáhla vysoké přesnosti 97 % na testovací databázi Ultrasonix, ale na druhé straně selhala na zbylých testovacích databázích s výsledky 29 % a 37 %.

Metoda LBP dosáhla vyrovnaných výsledků, ale je nevhodná pro detekci v reálném čase, jak bude popsáno dále v kapitole. Metoda Haar s trénovacím algoritmem k-NN dosáhla také vyrovnaných, ale podstatně horších výsledků než klasická kaskáda klasifikátorů založená pouze na metodě Haar, kde bylo dosaženo 97,39 % na

Tab. 4.14: Přesnost detekce pro databáze zdravých pacientů (omezené předpokladem výskytu pouze jedné tepny v obraze).

	Trénovací data Ultrasonix (283)	Testovací data Ultrasonix (538)	Testovací data Toshiba (433)
Kask. model	R_{TP} [%]	R_{TP} [%]	R_{TP} [%]
Haar	91,16	97,39	92,14
HOG	89,39	97,21	27,71
LBP	82,33	86,61	76,46
Haar & k-NN	85,86	97,76	80,36
Haar & evol.	96,46	97,58	95,15

Tab. 4.15: Přesnost detekce pro databáze nemocných pacientů (omezené předpokladem výskytu pouze jedné tepny v obraze).

	Trénovací data Philips (207)	Testovací data Philips (486)
Kask. model	R_{TP} [%]	R_{TP} [%]
Haar	99,51	91,44
HOG	29	36,65
LBP	96,61	95,11
Haar & k-NN	65,7	86,96
Haar & evol.	100	95,11

testovací databázi Ultrasonix, 92,14 % na databázi Toshiba a na testovací databázi Philips, kde byly artérie nemocných pacientů detekovány s přesností 91,44 %.

Z výsledků vyplývá, že vhodnější je použít detekční proces s funkcí zobrazující pouze jednu nejvýznamnější detekci. Dále je zřejmé, že kaskádní model založený na metodě Haar je možné použít i na snímcích z jiných přístrojů (Toshiba), než na kterých probíhalo trénování.

Evoluční optimalizace kaskádního modelu

Genetické programování, které je součástí evolučních algoritmů, bylo již úspěšně použito a publikováno autorem v práci [102], která se zabývala optimalizací výrobních procesů, a kde bylo dosaženo značné úspory provozních nákladů. Evoluční optimalizace byla zvolena i pro řešení problému detekce arterií.

Manuální nastavování parametrů kaskády klasifikátorů je náročný proces, kde každý řešený problém vyžaduje velmi specifické nastavení. Například pro trénování

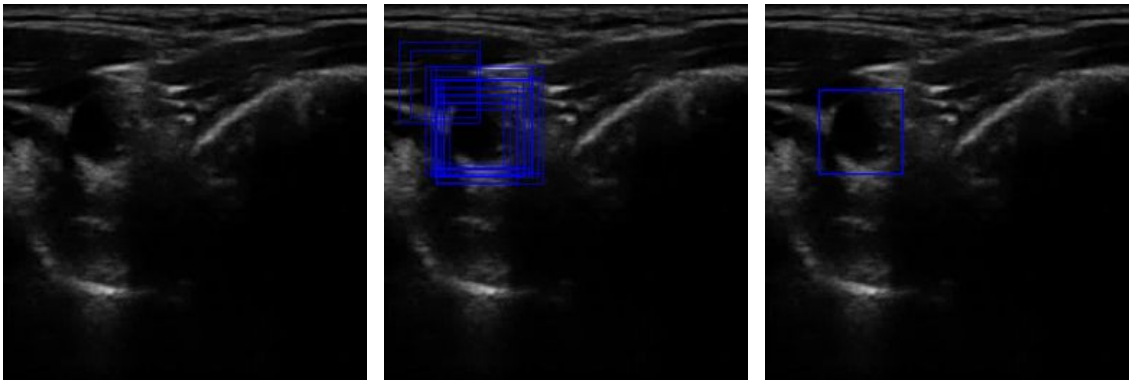
obličejového detektoru byla vytvořena kaskáda klasifikátorů o desítkách stupňů a tisících klasifikátorů. V případě detekce artérie bude celkové množství klasifikátorů i stupňů nižší. Evoluční optimalizace byla použita z důvodu nalezení vhodných parametrů kaskády klasifikátorů a za účelem zvýšení celkové detekční přesnosti. Na základě předchozích výsledků byla zvolena metoda Haar jako nejvhodnější kandidát pro natrénování finálního evolučně optimalizovaného kaskádního modelu.

Nejprve bylo s pomocí algoritmu AdaBoost vybráno 1000 nejlepších Haarových příznaků, které byly poté po převedení na lineární klasifikátory vstupem evoluční optimalizace. Druhým vstupem byla trénovací databáze Ultrasonix obsahující snímky s označenými pozicemi arterií, která sloužila pro validaci evolučně skládané kaskády klasifikátorů při výpočtu hodnoticí (fitness) funkce. Parametry evoluční optimalizace byly experimentálně nastaveny na 30 evolučních cyklů s velikostí populace 10 jedinců. Celý evoluční proces proběhl 10 krát vždy s jiným počtem klasifikátorů (100, 200, ...,1000). Kaskáda klasifikátorů s nejvyšší hodnotou hodnoticí funkce byla zvolena jako finální. Výsledná kaskáda klasifikátorů se skládá z 12-ti stupňů a z celkového počtu pouze 69-ti lineárních klasifikátorů, díky čemuž je detekční proces výpočetně mnohem méně náročný.

Tento výsledný model byl otestován na obrazových databázích a výsledky (Haar & evol.) jsou zobrazeny v již zmíněných tabulkách 4.12, 4.13, 4.14 a 4.15. Z výsledků je patrné že optimalizovaný model zlepšil stávající výsledky neoptimalizované metody Haar pro trénovací databázi Ultrasonix na 96,46 % (o 5,3 %), pro testovací databázi Toshiba na 95,15 % (o 3 %) a pro testovací databázi Philips (nemocní pacienti) na 95,11 % (o 3,67 %). Pro zbylé testované databáze byly výsledky srovnatelné. Celkově tedy původní metoda Haar dosahovala v průměru 94,32 % přesnosti detekce. Oproti tomu evolučně optimalizovaná metoda Haar dosahovala v průměru 96,86 % přesnosti detekce a tím bylo dosaženo zlepšení o 2,54 %.

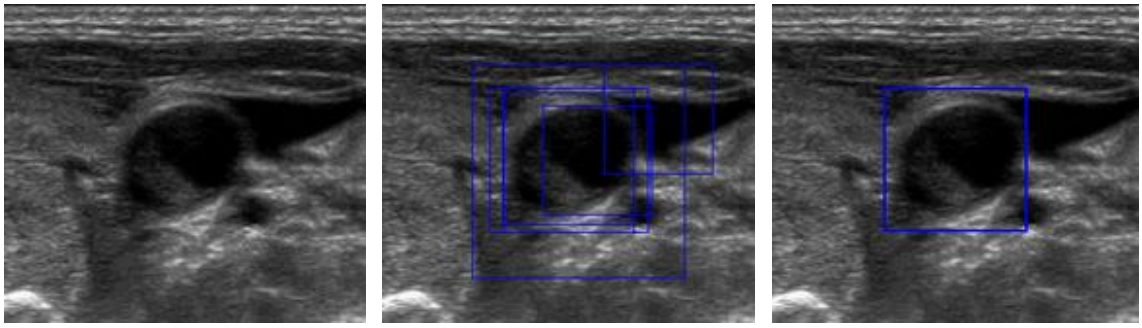
Detekované arterie

Výstupem detekčního procesu jsou lokalizované arterie v ultrazvukových snímcích. Detekční proces je zobrazen na obrázcích 4.18 a 4.19. Ve vstupním snímku zdravého pacienta (viz obrázek 4.18a) jsou hledány všechny výskyty arterie. Tyto výskyty jsou zobrazeny na obrázku 4.18b. Po následném sloučení a filtraci detekcí je výsledná detekce arterie zobrazena na obrázku 4.18c. Stejným způsobem je zpracován snímek nemocného pacienta (viz obrázek 4.19a), kde je detekovaná arterie zobrazena na obrázku 4.19c. Další výsledky detekovaných arterií jsou zobrazeny v příloze na obrázku A.3. Zobrazené nesprávné detekce arterií jsou v příloze na obrázku A.2.



(a) Originální snímek arterie (b) Zobrazení všech nalezených detekcí (c) Zobrazení detekované arterie

Obr. 4.18: Ukázka detekce zdravé arterie.



(a) Originální snímek arterie (b) Zobrazení všech nalezených detekcí (c) Zobrazení detekované arterie

Obr. 4.19: Ukázka detekce nemocné arterie.

Rychlost trénování a testování

Pro jednotlivé kaskádní modely byl měřen čas trénování a následně průměrný čas detekce jednoho snímku. Jednotlivé časy jsou uvedeny v tabulce 4.16. Časy trénování závisí na celkovém počtu daných příznaků a dále pak na výpočetní náročnosti jednotlivých typů příznaků. Například Haarovy příznaky jsou výpočetně nejméně náročné, ale na druhou stranu je jich pro zpracování více, než v případě zbývajících typů příznaků. Časy detekce závisí na výpočetní náročnosti jednotlivých příznaků a na struktuře kaskády klasifikátorů. První čtyři uvedené kaskádní modely mají stejnou strukturu (skládají se z 350 příznaků). Všechny uvedené kaskádní modely byly trénovány na jedné výše definované výpočetní stanici, pouze evoluční optimalizace při trénování modelu Haar & evol. byla provedena paralelně ve skupině 28-mi výpočetních stanic, kde byl dosažený čas 185 minut. V případě použití pouze jedné stanice by byl výpočet přibližně 28 krát pomalejší (asi 86 hodin).

Z tabulky vyplývá, že pro detekování objektů v reálném čase může být použita pouze metoda Haar s rychlostí detekce 4 ms na snímek (pro neoptimalizovaný model) a dále její optimalizovaná verze, která urychlila výpočet na 2,5 ms na snímek.

Tab. 4.16: Časová náročnost trénovacího a testovacího procesu.

Kask. model	Čas trénování [min]	Čas detekce [ms]
Haar	15,5	4
HOG	11,5	120
LBP	54	780
Haar & k-NN	15	95
Haar & evol.	185[4]	2,5

4.5.3 Srovnání s jinými pracemi

Přesná a efektivní lokalizace arterie z ultrazvukových snímků je dílčím krokem k diagnostikování možných kardio-vaskulárních nemocí. Mezi nejběžnější typ onemocnění, kde je důležitá přesná detekce arterie, patří ateroskleróza (kornatění tepen) [103]. Dále může být detekce arterie použita při diagnóze nemocí, jakými jsou ageneze arterie, hypoplazie [104] nebo aneurysma tepen [105].

Systém popsaný v rámci této práce byl porovnán (viz tabulka 4.17) s jinými řešeními, které se zabývají detekcí příčného řezu arterie z ultrazvukových snímků. Cílem vytvořené nové metody bylo natrénovat dostatečně robustní kaskádní model, který by měl vysokou přesnost detekce jak na zdravých, tak i nemocných pacientech.

Tab. 4.17: Porovnání současných řešení pro detekci příčného řezu arterie.

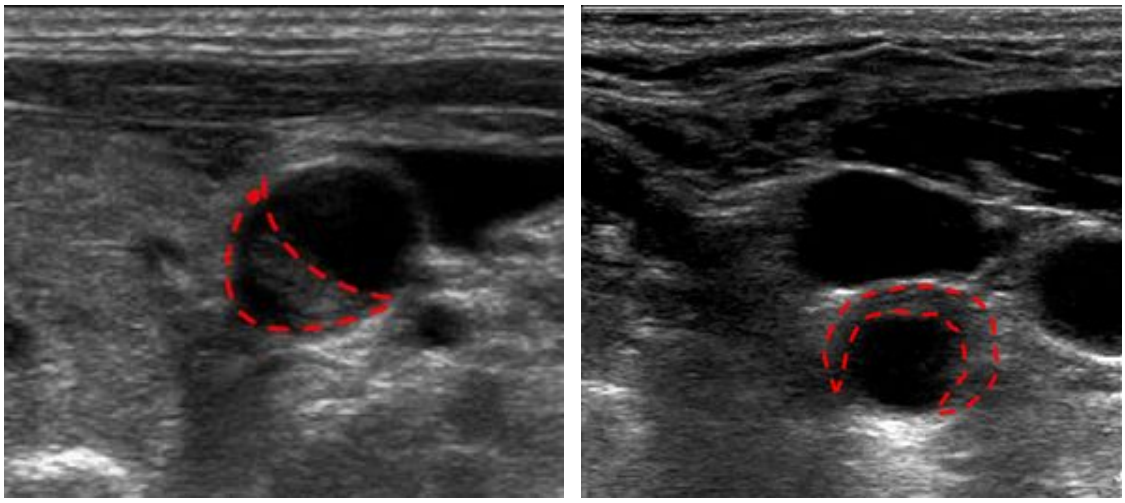
	Zdraví pac.			Nemocní pac.
Řešení	R_{TP} [%]	Detekce [s]	Metoda	R_{TP} [%]
Golemati a kol. [106]	83	0,166	HT[5]	54
Říha a Beneš [107]	84	3	HT[5], OT[6]	—
Beneš a kol. [108]	90	1,5	HT[5], OT[6]	—
Říha a kol. [1]	**97**	0,013	Viola-Jones	57
Nová metoda	96	**0,0025**	Viola-Jones	**95**

[4]Při použití 28-mi výpočetních stanic.
[5]Houghova transformace.
[6]Optický tok.

Měření úspěšnosti jednotlivých řešeních probíhalo na základě srovnání přesnosti detekce na zdravých i nemocných pacientech, dále byl také porovnáván čas zpracování jednoho snímku. Vzhledem k tomu, že každé řešení využívá jiné databáze testovacích snímků i jiné výpočetní zařízení, jsou tyto výsledky spíše orientační, než že by se jednalo o přesné srovnání jednotlivých metod. Z výsledků vyplývá, že nová metoda vytvořená v rámci této práce dosáhla druhé nejvyšší přesnosti detekce na zdravých (96 %) databázích pacientů a dále nejvyšší přesnosti detekce na nemocných (95 %) databázích pacientů. Průměrná rychlost detekce jednoho snímku byla 2,5 ms. Při srovnání použitých metod dosáhla metoda Viola-Jones (kaskádní detektor s Haarovými příznaky) oproti metodám optického toku a Houghovy transformace, jak vyšší přesnosti detekce, tak i vyšší rychlost detekce.

4.6 Určení stupně plaku v arterii

Určením množství plaku v arterii lze snadněji předpovídat nemoc, jakou je například ateroskleróza. Na obrázku 4.20a je v části arterie vyznačena nižší úroveň plaku. Dále pak na obrázku 4.20b je zobrazena rozsáhlejší úroveň plaku. V předcházející kapitole byl otestován kaskádní model, který dokáže přesně lokalizovat arterii v ultrazvukovém snímku. Následným cílem je tuto detekovanou artérii klasifikovat do některé ze tříd: zdravá, nízký stupeň plaku, vyšší stupeň plaku.



(a) Pacient s nižší úrovní plaku. (b) Pacient s vyšší úrovní plaku.

Obr. 4.20: Označení plaku v arterii.

4.6.1 Dosažené výsledky

Pro klasifikaci detekované arterie byl natrénován klasifikační model hlubokého učení. Pro trénování byla vytvořena nová trénovací databáze snímků skládající se z celkem 300 snímků, kde 100 snímků zdravých pacientů bylo vybráno z trénovací databáze Ultrasonix, z trénovací databáze Philips bylo dále vybráno 100 snímků nemocných pacientů s nižší úrovní plaku a 100 snímků nemocných pacientů s vyšší úrovní plaku.

Tato databáze byla vstupem konvoluční neuronové sítě, která sloužila k natrénování klasifikačního modelu. Celkové nastavení všech parametrů konvoluční sítě je velmi rozsáhlé a složité. Proto byla provedena celá řada experimentů, kde byly měněny počty a typy jednotlivých vrstev, typy výstupních výpočetních funkcí nebo rozměry vstupních snímků a počet trénovacích iterací. Po sérii takto provedených experimentů byla dosažena dostačující přesnost (viz tabulka 4.18) pro výslednou konfiguraci konvoluční neuronové sítě:

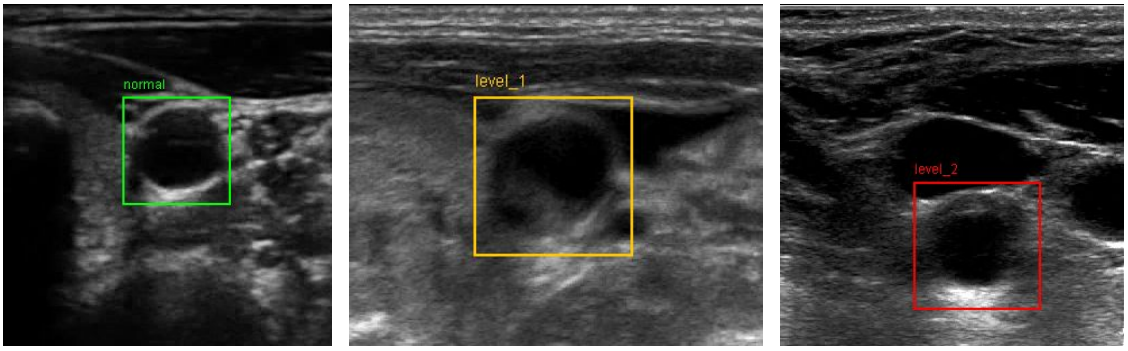
- jedna konvoluční vrstva,
- jedna pod vzorkovací vrstva,
- výstupní plně propojená vrstva se třemi výstupy,
- rozměry vstupních snímků - 24x24 pixelů,
- počet iterací - 30.

Přesnost klasifikace R_{TP} byla spočtena jako podíl správně klasifikovaných snímků vůči všem snímkům v dané databázi. Celková průměrná dosažená přesnost klasifikace byla 98 %. Doba trénování finálního klasifikačního modelu byla 15 minut. Průměrný změřený čas detekce a klasifikace jednoho snímku byl 25 ms, což znamená, že vytvořený systém je schopný pracovat v reálném čase.

Tab. 4.18: Přesnost klasifikace pro jednotlivé databáze snímků.

Databáze	R_{TP} [%]
Ultrasonix - trénovací (283)	99,7
Ultrasonix - testovací (538)	98,67
Toshiba - testovací (433)	96,6
Philips - trénovací - stupeň 1 (105)	99,05
Philips - trénovací - stupeň 2 (102)	100
Philips - testovací - stupeň 1 (385)	97,78
Philips - testovací - stupeň 2 (101)	100

Na obrázcích 4.21a, 4.21b a 4.21c jsou zobrazeny příklady detekovaných a klasifikovaných arterií. Každá detekce arterie byla vykreslena jinou barvou podle typu



(a) Snímek zdravého pacienta

(b) Snímek nemocného pacienta s nižší úrovní plaku.

(c) Snímek nemocného pacienta s vyšší úrovní plaku.

Obr. 4.21: Ukázka klasifikace arterie.

klasifikační třídy. Dále jsou v příloze na obrázcích A.3 a A.4 správně detekovány a klasifikovány arterie jak zdravých, tak nemocných pacientů. Nesprávné klasifikace arterií jsou uvedeny v příloze na obrázku A.5.

Ačkoliv průměrný výsledek 98 % je zcela dostačující, je nutné zmínit, že z důvodů nízkého množství dat nemocných pacientů (je obtížné tato data získat), byla konvoluční neuronová síť natrénována spíše na rozpoznání těchto konkrétních dvou pacientů, než na obecnou klasifikaci pacientů s nižším a vyšším obsahem plaku v arteriích. Při použití většího souboru dat pacientů s různým stupněm nemoci by byl natrénovaný klasifikační model výrazně robustnější a použitelnější v praxi.

5 DISKUSE VÝSLEDKŮ

V předcházející kapitole byly na vybraných příkladech provedeny testy, které deklarovaly funkčnost vytvořených systémů. V této kapitole bude shrnuto srovnání výsledků s jinými pracemi a výhody či nevýhody vytvořených řešení.

5.1 Srovnání s jinými pracemi

Současné dosažené výsledky byly porovnány s novými přístupy, které řešily podobné problémy. Dosažené výsledky byly pro řešené problémy ve většině případů přesnější při snížené časové výpočetní náročnosti. Detekcí domů z leteckých snímků se zabývaly práce [94, 95], kde nejvyšší přesnosti detekce bylo dosaženo v [95] (95 %) ve srovnání s navrženým postupem (78 %), který byl ale navíc optimalizován pro rychlý výpočet. Předností nového postupu je tedy rychlost vyhodnocení obrazových dat oproti předchozím metodám. Vzhledem k rozdílným typům dat není dosažení horšího výsledku přesnosti plně vypovídající.

Srovnání učícího se algoritmu k-NN optimalizovaného pro grafické akcelerátory bylo provedeno s pracemi [63, 97, 96, 98]. Vytvořená optimalizovaná metoda dosáhla 882-ti násobného zrychlení (oproti běžné CPU verzi) a v porovnání s uvedenými pracemi, bylo toto dosažné zrychlení nejvyšší. Zároveň pouze zde byla podpora pro použití více akcelerátorů současně. Druhé nejvyšší zrychlení bylo dosaženo v [98] (336-ti násobné). Uvedené práce používaly pro výpočty speciální verze výpočetního hardware Nvidia Tesla. Oproti tomu vytvořená metoda využívala běžně dostupnou herní grafickou kartu. Výpočty, které na CPU trvají běžně dny, jsou v případě použití GPU provedeny v několika minutách.

V případě trénování optimalizované verze objektového detektoru pro GPU bylo provedeno pouze srovnání s prací [101], protože ostatní práce se většinou zabývají optimalizací detekčního procesu a neřeší již optimalizaci trénování. V uvedené práci [101] byl použit výkonný profesionální grafický akcelerátor Nvidia Tesla a pro jedno GPU bylo dosažené zrychlení 34 násobné. Nová metoda využívající běžný grafický akcelerátor oproti tomu dosáhl vyššího zrychlení (137-mi násobné) s použitím čtyř paralelně běžících GPU.

Detekce příčných řezů arterie z ultrazvukových snímků byla řešena v rámci prací [106, 108, 107], kde byly použity metody založené na Houghově transformaci případně dále metoda optického toku. Tyto přístupy nepřesáhly 90 % přesnosti detekce a ani rychlost detekce neprobíhala v reálném čase. Původní metoda zveřejněná v impaktovaném článku [1] byla založena na objektovém detektoru (metoda Viola-Jones), kde dosažená přesnost na datech zdravých pacientů byla 97 %. Nově navržená

metoda rozšiřuje a zároveň vylepšuje výsledky publikované v tomto impaktovaném článku, kdy pro databáze nemocných pacientů byla dosažena přesnost 95 % oproti původním 57 % při rychlosti detekce 2,5 ms na snímek. Pro databáze zdravých pacientů byla dosažena přesnost 96 %. Dále byl tento detekční systém rozšířen o možnosti klasifikace detekované arterie do kategorií podle úrovně plaku obsaženého v arterii. Průměrný dosažený výsledek klasifikace byl 98 %, ale z důvodů nízkého počtu nemocných pacientů není výsledek tolik vypovídající.

5.2 Výhody a nevýhody vytvořeného řešení

Jednotlivé vytvořené části systému mají své výhody i nevýhody. U algoritmu trénovatelné segmentace je výhodou snadné označení částí určených k trénování. Další výhodou je vyšší rychlost zpracování díky navržené optimalizaci nebo možnost použít množství libovolných algoritmů pro obrazové transformace. V současné době je nevýhodou klasifikace členění pouze do dvou tříd.

U optimalizované verze algoritmu k-NN je výhodou jeho velmi vysoká rychlost zpracování větších objemů dat v porovnání s klasickou CPU verzí algoritmu. Nevýhodou je nevhodnost algoritmu pro řešení složitějších problémů. To byl také důvod, proč nebyl tento algoritmus použit pro řešení klasifikačních problémů popsaných ve výsledcích této práce.

Výhodou systému pro detekci objektů v obrazech je jeho komplexnost, která spočívá v možnostech výběru více typů příznaků, které je možné aplikovat pro řešení různých typů problémů. Další možností je výběr typu výstupního modelu (kaskáda klasifikátorů, nebo kaskáda modelů umělé inteligence), kdy první přístup je rychlejší a druhý přístup využívá různé kombinace učících se algoritmů, s kterými lze při řešení určitých problémů dosáhnout přesnějších výsledků. Mezi výhody patří optimalizace výpočetně náročných funkcí, které jsou optimalizovány, jak pro CPU, tak i pro GPU. Díky evoluční optimalizaci je možné automaticky natrénovat model s vyšší přesností detekce, než v případě ručního zvolení parametrů tohoto modelu. Dále detekční systém umožňuje provést následnou klasifikaci detekované oblasti podle přetrénovaného modelu hlubokého učení. Mezi nevýhody patří velmi pomalá funkčnost při provádění operací s příznaky LBP. Dále jeden detektor může být použit pro detekci jednoho typu objektů, tudíž pokud by bylo třeba provést klasifikaci detekovaných částí do více různorodých tříd, musel by být pro každou třídu natrénován vlastní detektor.

Obecně lze za výhodu považovat používání algoritmů optimalizovaných pro GPU. Tyto algoritmy oproti běžným CPU verzím těchto algoritmů výrazně šetří elektrickou energii a výpočetní čas.

6 ZÁVĚR

Práce se zabývá optimalizovanými automatickými metodami strojového učení pro získávání znalostí z multimediálních dat. V dnešní době je vývoj optimalizovaných verzí současných algoritmů i vývoj nových algoritmů potřebný z důvodu nárůstu objemů multimediálních dat, která je nutné kvalitně a efektivně zpracovávat a to většinou v reálném čase. Díky speciálním typům paralelního hardware byly výpočty výrazně urychleny a algoritmus mohl být otestován vícekrát s různými parametry za účelem dosažení větší přesnosti při řešení vybraných příkladů. Výpočty trvající běžně v řádech dní bylo možné urychlit navrženými metodami na několik málo minut a zároveň tím byla ušetřena elektrická energie.

Vlastní řešení práce se věnuje nejprve rozboru stávajícího stavu problematiky, kde jsou probrány současné metody pro detekci objektů v obrazech, dále učící se algoritmy umělé inteligence včetně velmi nové problematiky hlubokého učení. Nakonec byla popsána metoda trénovatelné segmentace, která kombinuje techniky pro zpracování obrazu s učícími se algoritmy umělé inteligence.

V další části bylo navrženo řešení, které se zabývá návrhem a optimalizací metod pro získávání znalostí z multimediálních dat. Nejprve byl navržen komplexní systém pro trénování a testování objektového detektoru. Trénování bylo rozšířeno oproti původnímu přístupu [7] o další typy obrazových příznaků (HOG, LBP). Dále byla tato část optimalizována pro zpracování na grafických akcelerátorech s pomocí prostředí CUDA za účelem rychlého natrénování detektoru na velkých databázích obrazů. Pro dosažení vyšší přesnosti lze parametry natrénovaného detektoru dále optimalizovat s pomocí evolučních algoritmů. Testovací část procesu byla rozšířena o možnosti klasifikace detekovaných oblastí do tříd podle natrénovaného modelu hlubokého učení. Dále byla provedena optimalizace učících se algoritmů umělé inteligence, kde byla u několika algoritmů paralelizována testovací část pro nasazení na běžném procesoru. Pro algoritmus k-nejbližších sousedů byla provedena optimalizace pro grafické akcelerátory, kde byl pro tento algoritmus výrazně změněn výpočetní princip oproti původní procesorové verzi. Pro programování algoritmů pro GPU bylo vytvořeno speciální rozhraní, které usnadňovalo vývoj těchto typů algoritmů a dále umožňovalo automatické distribuování výpočtů mezi více grafických akcelerátorů. Pro metodu trénovatelné segmentace byla optimalizována testovací část, aby bylo možné výpočet efektivně spouštět na více jádrových procesorech. Všechny implementované algoritmy byly vytvořeny v programovacím jazyku JAVA, výjimkou jsou pouze kódy vytvořených GPU algoritmů, které jsou psané v jazyce C s využitím funkcí z prostředí CUDA a OpenCL.

Hlavním přínosem této práce je výzkum, implementace, experimentální ověření a optimalizace automatizovaných strojových metod pro dolování znalostí z multi-

mediálních dat, kde na vybraných případech bylo s jejich pomocí dosaženo vyšší přesnosti v porovnání s konvenčními metodami. Vybrané výsledky byly publikovány v časopisech s impaktním faktorem [1, 2]. Tyto metody mohou využívat speciální paralelní hardware, díky kterému je dosaženo úspory elektrické energie i výpočetního času při dosažení lepší přesnosti řešených problémů. Vytvořené metody jsou přístupné v rámci jednoho komplexního systému, kde mohou být jednoduše použity a kombinovány.

Funkčnost vytvořeného systému byla ověřena na vybraných příkladech. Prvotní verze detekce arterií z ultrazvukových snímků byla publikována v impaktovaném časopise [1] (IF=2,1) a v rámci této práce byla rozšířena o natrénování nového detektoru, který funguje jak na zdravých, tak i nemocných pacientech. S pomocí evoluční optimalizace byl tento objektový detektor otestován na nově vytvořených databázích, kde bylo při zpracování v reálném čase (2,5 ms na snímek) dosaženo 96 % úspěšnosti detekce pro data zdravých pacientů a 95 % úspěšnosti detekce pro data nemocných pacientů. Tato evoluční optimalizace byla nasazena v klastru počítačů, kdy výpočet trval 185 minut při použití 28-mi výpočetních stanic oproti 86 hodinám v případě použití jedné výpočetní stanice. Bez použití evoluční optimalizace byla přesnost detekce modelu v průměru o 2,5 % nižší. Dále byl pro každou detekovanou arterii klasifikován stupeň její nemoci s pomocí algoritmů hlubokého učení, kdy bylo dosaženo 98 % přesnosti klasifikace pro 3 úrovně plaku v arterii. Dalším řešeným příkladem byla detekce domů ze satelitních snímků. S pomocí metody trénovatelné segmentace byla dosažena přesnost detekce 78 %. Za použití metody trénovatelné segmentace byly z 3D obrazů počítačové tomografie detekovány materiály obsažené ve třech meteoritech. Výsledkem celého procesu bylo spočtené zastoupení těchto materiálů, kde 2,4 % objemu meteoritů bylo tvořeno kovy (železo, nikl) a 2,9 % tvořily sulfidy (troilit).

Jedním z dalších přínosů je vytvoření a otestování optimalizovaných algoritmů navržených pro grafické akcelerátory. Algoritmus k-nejbližších sousedů dosáhl 882-ti násobného zrychlení v porovnání s běžnou verzí algoritmu. Při provedených testech bylo pro verzi pro GPU spotřebováno 66x méně elektrické energie. V tomto případě probíhalo porovnání spotřeby mezi vytvořeným optimalizovaným algoritmem a odpovídajícím algoritmem z prostředí programu RapidMiner. Protože jsou grafické akcelerátory používané v dnešní době v porovnání s použitou Nvidia GeForce GTX 690 podstatně výkonnější a zároveň energeticky úspornější díky snižujícím se rozměrům vyráběné architektury jádra GPU, bude hodnota ušetřené elektrické energie ještě vyšší. Při srovnání cen je hardwarové řešení skládající se z grafických akcelerátorů v tomto případě 59x levnější, než při použití klasických výpočetních stanic s CPU. Při testování optimalizovaného algoritmu určeného pro trénování objektového detektoru bylo dosaženo 137-mi násobného zrychlení, při 10,5x nižší spotřebě

elektrické energie. Cena řešení s grafickým hardwarem by byla při stejném výkonu v tomto případě 10x levnější.

Klasifikace hutních materiálů pomocí laserové spektrometrie [5] (IF=3,047, zatím odesláno k publikování) byla provedena s použitím nově navržených postupů pro zpracování obrazu a učících se algoritmů umělé inteligence. Výsledky ukázaly, že oproti běžným postupům, lze výrazně zvýšit celkovou přesnost na 99,1 % při klasifikaci materiálů do 50 tříd. Automatické rozpoznávání emocí z textu bylo publikováno v impaktovaném časopise [2] (IF=0,59), kde byla dosažena přesnost 86,89 % při klasifikaci do pěti emočních tříd, přičemž bylo dosaženo 11,4 % zlepšení oproti současným řešením.

Na práci by mohlo být v budoucnu navázáno vytvořením dalších optimalizovaných algoritmů umělé inteligence. V případě procesu trénovatelné segmentace by mohla být práce rozšířena o možnosti pracovat s více klasifikačními třídami.

LITERATURA

[1] ŘÍHA, K.; MAŠEK, J.; BURGET, R.; BENEŠ, R.; ZÁVODNÁ, E. "Novel method for localization of common carotid artery transverse section in ultrasound images using modified Viola–Jones detector." *Ultrasound in Medicine & Biology.* 2013, s. 1887-1902, ISSN 0301-5629.

[2] POVODA, L.; BURGET, R.; MASEK, J.; UHER, V.; DUTTA, M. K. "Optimization Methods in Emotion Recognition System" *Radioengineering [online],* 2016.

[3] MAŠEK, J.; BURGET, R.; POVODA, L.; DUTTA, M. "Multi–GPU Implementation of Machine Learning Algorithm using CUDA and OpenCL." *International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems*, 2016, roč. 5, č. 2, s. 101-107. ISSN: 1805-5443.

[4] MAŠEK, J.; BURGET, R.; KARÁSEK, J.; UHER, V.; DUTTA, M. "Multi–GPU Implementation of k-Nearest Neighbor Algorithm," *In Proceedings of the 38th International Conference on Telecommunication and Signal Processing.* Berlin, Germany: 2014. s. 764-767. ISBN: 978-1-4799-8497-8.

[5] KLUS J., PORIZKA P., MASEK J., RAJNOHA M., BURGET R., PROCHAZKA D., SKARKOVA P., SLADKOVA L., NOVOTNY J., KAISER J. "Multivariate classification of echellograms - new perspective in Laser-Induced Breakdown Spectroscopy analysis.", *Spectrochimica Acta Part B: Atomic Spectroscopy,* 2016, **(odesláno k publikování)** (IF: 3.047)

[6] BURGET, R.; UHER, V.; MAŠEK, J. "Trainable Segmentation Based on Local-level and Segment-level Feature Extraction," *In IEEE International Symposium on Biomedical Imaging,* Barcelona: 2012. s. 17-24. ISBN: 978-1-4673-1118-2.

[7] VIOLA, P. a M. JONES. "Rapid Object Detection using a Boosted Cascade of Simple Features," *In: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition.* CVPR 2001. IEEE Comput. Soc, 2001, s. I-511-I-518, ISBN 0-7695-1272-0.

[8] MAŠEK, J.; BURGET, R.; UHER, V. "IMMI: Interactive Segmentation Toolkit," *In Engineering Applications of Neural Networks. Communications in Computer and Information Science.* Heidelberg: Springer Berlin Heidelberg, 2013. s. 380-387. ISBN: 978-3-642-41012-3, ISSN: 1865-0929.

[9] KALASOVA D., MASEK J., ZIKMUND T., SPURNY P., BURGET R., KAISER J. "Segmentation of multiphase materials applying trainable segmentation.", *7th Conference on Industrial Computed Tomography, Leuven, Belgium (iCT 2017),* 2017, **(odeslaný k publikování)**.

[10] WU, Jianxin, Christopher GEYER a James M. REHG. "Real-time human detection using contour cues,"*In: 2011 IEEE International Conference on Robotics and Automation.* IEEE, 2011, s. 860-867, ISBN 978-1-61284-386-5.

[11] JEONG, Mira, Byoung Chul KO a Jae-Yeal NAM "Early Detection of Sudden Pedestrian Crossing for Safe Driving during Summer Nights." *EEE Transactions on Circuits and Systems for Video Technology.* 2016. ISBN: 1051-8215.

[12] ZAKI, Mohamed H. a Tarek SAYED "Exploring walking gait features for the automated recognition of distracted pedestrians." *IET Intelligent Transport Systems* 2016. s. 106-113. DOI: 10.1049/iet-its.2015.0001.

[13] SATZODA, Ravi Kumar a Mohan Manubhai TRIVEDI "Multipart Vehicle Detection Using Symmetry-Derived Analysis and Active Learning." *IEEE Transactions on Intelligent Transportation Systems* 2016. s. 926-937. ISSN 1524-9050.

[14] SEENOUVONG, Nilakorn, Ukrit WATCHAREERUETAI, Chaiwat NUTHONG, Khamphong KHONGSOMBOON a Noboru OHNISHI "A computer vision based vehicle detection and counting system." *In: 2016 8th International Conference on Knowledge and Smart Technology (KST)* 2016. s. 224-227. ISBN 978-1-4673-8137-6.

[15] NOVOTNY D. a Matas J. "Cascaded Sparse Spatial Bins for Efficient and Effective Generic Object Detection." *2015 IEEE International Conference on Computer Vision (ICCV), Santiago* 2015. s. 1152-1160. DOI: 10.1109/ICCV.2015.137

[16] GIRSHICK R. "Fast R-CNN." *2015 IEEE International Conference on Computer Vision (ICCV), Santiago* 2015. s. 1440-1448. DOI: 10.1109/ICCV.2015.169

[17] FREUND, Yoav a Robert E SCHAPIRE. "A decision-theoretic generalization of on–line learning and an application to boosting."*In Journal of Computer and System Sciences.* 1997, ISSN 00220000

[18] LIENHART, R. a J. MAYDT. "An Extended Set of Haar–like Features for Rapid Object Detection." *In: Proceedings. International Conference on Image Processing* IEEE, 2002, 2015-11-20, s. I-900-I-903, ISBN 0-7803-7622-6.

[19] CHANG HUANG, HAIZHOU AL, BO WU a SHIHONG LAO. "Boosting Nested Cascade Detector for Multi-View Face Detection." *In: Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.* s. 415-418 Vol.2, ISBN 0-7695-2128-2.

[20] VIOLA, P. a M. JONES. "Fast Multi-view Face Detection."*Proc. of Computer Vision and Pattern Recognition*. Mitsubishi Electric Research Lab, 2003

[21] YING, Ying, Han WANG a Jian XU. "An automatic system for multi-view face detection and pose estimation."*In: 2010 11th International Conference on Control Automation Robotics & Vision*. IEEE, 2010, s. 1101-1108, ISBN 978-1-4244-7814-9.

[22] JUN-SU JANG a JONG-HWAN KIM. "Fast and Robust Face Detection Using Evolutionary Pruning. " *IEEE Transactions on Evolutionary Computation*. 2008, s. 562-571, ISSN 1941-0026.

[23] WEIWEI ZHANG, JIAN SUN a XIAOOU TANG. "From Tiger to Panda: Animal Head Detection." *IEEE Transactions on Image Processing*. 2011, s. 1696-1708, ISSN 1057-7149.

[24] BURGHARDT, Tilo a Janko CALIC. "Real-time Face Detection and Tracking of Animals"*In: 2006 8th Seminar on Neural Network Applications in Electrical Engineering* . 2006, s. 27-32, ISBN 1-4244-0432-0.

[25] LIU YUN; ZHANG PENG. "An Automatic Hand Gesture Recognition System Based on Viola-Jones Method and SVMs"*Second International Workshop on Computer Science and Engineering*. 2009, s. 72-76, ISBN 978-0-7695-3881-5

[26] PUTRO, M. Dwisnanto, Teguh Bharata ADJI a Bondhan WINDURATNA. "Adult image classifiers based on face detection using Viola-Jones method." *In: 2015 1st International Conference on Wireless and Telematics (ICWT)* 2015. s. 1-6. ISBN 978-1-4673-8433-9.

[27] RAY, Sambarta, Souvik DAS a Anindya SEN "An intelligent vision system for monitoring security and surveillance of ATM." *In: 2015 Annual IEEE India Conference (INDICON)* IEEE, 2015, s. 1-5. ISBN 978-1-4673-7399-9.

[28] OMIDYEGANEH, Mona, Shervin SHIRMOHAMMADI, Shabnam ABTAHI, et al. "Yawning Detection Using Embedded Smart Cameras." *IEEE Transactions on Instrumentation and Measurement* 2016. s. 570-582. ISSN 0018-9456.

[29] NVIDIA Inc, "CUDA Toolkit 7.5."2015, Available: `https://developer.nvidia.com/about-cuda`

[30] ORO, David, Carles FERNANDEZ, Javier Rodriguez SAETA, Xavier MARTORELL a Javier HERNANDO. "Real–time GPU-based face detection in HD video sequences."*In: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops).* IEEE, 2011, s. 530-537, ISBN 978-1-4673-0063-6.

[31] SHARMA, Bharatkumar, Rahul THOTA, Naga VYDYANATHAN a Amit KALE. "Towards a robust, real–time face processing system using CUDA-enabled GPUs" *In: 2009 International Conference on High Performance Computing (HiPC).* IEEE, 2009, s. 368-377, ISBN 978-1-4244-4922-4.

[32] KONG, Jiangang a Yangdong DENG. "GPU accelerated face detection."*Intelligent Control and Information Processing (ICICIP),* IEEE, 2010, s. 584-588, ISBN 978-1-4244-7047-1.

[33] JIA, Haipeng, Yunquan ZHANG, Weiyan WANG a Jianliang XU. "Accelerating Viola–Jones Facce Detection Algorithm on GPUs,"*In: 2012 IEEE 14th International Conference on High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems.* IEEE, 2012, s. 396-403, ISBN 978-1-4673-2164-8.

[34] ACASANDREI, Laurentiu a Angel BARRIGA. "Accelerating Viola–Jones face detection for embedded and SoC environments." *In: 2011 Fifth ACM/IEEE International Conference on Distributed Smart Cameras.* IEEE, 2011, s. 1-6, ISBN 978-1-4577-1708-6.

[35] BURGET, R.; CIKA, P.; ZUKAL, M.; MAŠEK, J. "Automated Localization of Temporomandibular Joint Disc in MRI Images."*In 2011 34th International Conference on Telecommunications and Signal Processing (TSP).* 2011, s. 413-416, ISBN: 978-1-4577-1409-2.

[36] MAŠEK, J.; BURGET, R.; KARÁSEK, J.; UHER, V.; GÜNEY, S. "Evolutionary Improved Object Detector for Ultrasound Images."*In 2013 36th International Conference on Telecommunications and Signal Processing (TSP).* 2013, s. 586-590, ISBN: 978-1-4799-0402-0.

[37] MAŠEK, J.; BURGET, R.; UHER, V.; GÜNEY, S. "Speeding up Viola–Jones algorithm using multi–Core GPU implementation."*In 2013 36th International Conference on Telecommunications and Signal Processing (TSP).* 2013, s. 808-812, ISBN: 978-1-4799-0402-0.

[38] KARÁSEK, J.; BURGET, R.; MAŠEK, J.; BENDA, O. "Genetic Programming Based Classifier in Viola-Jones RapidMiner Image Mining Extension,"*In 2013*

36th International Conference on Telecommunications and Signal Processing (TSP). 2013, s. 872-876. ISBN: 978-1-4799-0402-0.

[39] MAŠEK, J. "Detekce objektů v obraze s pomocí Haarových příznaků." *In Proceedings of the 18th Conference STUDENT EEICT 2012 Volume 2.* 2012. s. 70-72. ISBN: 978-80-214-4461-4.

[40] BALLARD, D. H.; BROWN, Ch. M. "Computer Vision."*Englewood Cliffs, N.J.: Prentice Hall* 1982. ISBN: 978-0131653160

[41] FORSYTH, D; PONCE J. "Computer vision: a modern approach." *2nd ed. Boston: Pearson* 2012. ISBN: ISBN 9780136085928.

[42] DALAL, N. a TRIGGS B. "Histograms of Oriented Gradients for Human Detection."*2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05).* IEEE, 2005, č. 1, s. 886-893. DOI: 10.1109/CVPR.2005.177.

[43] MOHAN, A., C. PAPAGEORGIOU a T. POGGIO. "Example-based object detection in images by components."*IEEE Transactions on Pattern Analysis and Machine Intelligence.* 2002, svazek 23, vydání 4, s. 349-361. DOI: 10.1109/34.917571.

[44] QIANG ZHU, MEI-CHEN YEH, KWANG-TING CHENG a S. AVIDAN. "Fast Human Detection Using a Cascade of Histograms of Oriented Gradients."*2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2 (CVPR'06).* IEEE, 2006, č. 2, s. 1491-1498. DOI: 10.1109/CVPR.2006.119.

[45] JIA, Hui-Xing a Yu-Jin ZHANG. "Fast Human Detection by Boosting Histograms of Oriented Gradients."*In: Fourth International Conference on Image and Graphics (ICIG 2007).* IEEE, 2007, s. 683-688, ISBN 978-0-7695-2929-5.

[46] DALAL, Navneet, Bill TRIGGS a Cordelia SCHMID "Human Detection Using Oriented Histograms of Flow and Appearance."*In Computer Vision – ECCV 2006* s. 428, DOI: 10.1007/11744047_33.

[47] BERTOZZI, M., A. BROGGI, M. Del ROSE, M. FELISA, A. RAKOTOMAMONJY a F. SUARD. "A Pedestrian Detector Using Histograms of Oriented Gradients and a Support Vector Machine Classifier."*In: 2007 IEEE Intelligent Transportation Systems Conference.* IEEE, 2007, s. 143-148, ISBN 978-1-4244-1395-9.

[48] SUARD, F., A. RAKOTOMAMONJY, A. BENSRHAIR a A. BROGGI. "Pedestrian Detection using Infrared images and Histograms of Oriented Gradients."*In: 2006 IEEE Intelligent Vehicles Symposium.* IEEE, 2006, s. 206-212. DOI: 10.1109/IVS.2006.1689629.

[49] CHEN, Ziyi, Cheng WANG, Chenglu WEN, et al. "Vehicle Detection in High-Resolution Aerial Images via Sparse Representation and Superpixels." *IEEE Transactions on Geoscience and Remote Sensing* 2016. s. 103-116. ISSN 0196-2892.

[50] HUANG, Zhiyong, Yuanlong YU, Jason GU a Huaping LIU "An Efficient Method for Traffic Sign Recognition Based on Extreme Learning Machine." *IEEE Transactions on Cybernetics* 2016. s. 1-14. ISSN 2168-2267.

[51] GONZALEZ, Rafael C a Richard E WOODS. "Digital image processing. " *3rd ed. Upper Saddle River, N.J.: Pearson Prentice Hall* 2008. ISBN 978-0-13-505267-9.

[52] SZELISKI, Richard. "Computer Vision : Algorithms and Applications."*New York: Springer.*, 2011. 811 s. ISBN 978-1-84882-934-3.

[53] AGADA, Ruth a Jie YAN. "Edge based mean LBP for valence facial expression detection."*In: 2015 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)* 2015.s 1-7. ISBN 978-1-4799-6084-2.

[54] MAMMERI, Abdelhamid, Depu ZHOU a Azzedine BOUKERCHE "Animal-Vehicle Collision Mitigation System for Automated Vehicles."*EEE Transactions on Systems, Man, and Cybernetics: Systems* 2016. s 1-13. ISSN 2168-2216.

[55] MÄENPÄÄ, T. "The local binary pattern approach to texture analysis: extenxions and applications."*Oulu: Oulun yliopisto* 2003. ISBN 9514270762

[56] POVODA, L.; BURGET, R.; MAŠEK, J.; CVRK, L. "Automatické rozpoznávanie emócií z českého textu pomocou umelej inteligencie." *Elektrorevue - Internetový časopis (http://www.elektrorevue.cz)*, 2015, roč. 17, č. 1, s. 15-18. ISSN: 1213-1539.

[57] UHER, V.; BURGET, R.; MAŠEK, J.; KARÁSEK, J. "Audio mining extension." *In RapidMiner Community Meeting And Conference - RCOMM 2013.* 2013. s. 12-18. ISBN: 978-3-8440-2145-5.

[58] MAŠEK, J.; BURGET, R.; POVODA, L.; HARVÁNEK, M. "Image Search Using Similarity Measures Based on Circular Sectors." *In Computer Science &*

Information Technology (CS & IT). Dubaj, Spojené arabské emiráty: 2015. s. 241-251. ISBN: 978-1-921987-43-4. ISSN: 2231-5403.

[59] SINGH, A.; DUTTA, M.; BURGET, R.; MAŠEK, J. "Identification of Acrylamide in Fried Potato Crisps Using Image Processing in Wavelet Domain." *In 2015 38th International Conference on Telecommunications and Signal Processing (TSP).* Prague, Czech Republic: 2015. s. 545-549. ISBN: 978-1-4799-8497-8. ISSN: 1805-5435.

[60] KARÁSEK, J.; BURGET, R.; UHER, V.; MAŠEK, J.; DUTTA, M. "Color Image (Dis) Similarity Assessment and Grouping based on Dominant Colors." *In 2014 37th International Conference on Telecommunications and Signal Processing (TSP).* Berlin, Germany: 2014. s. 631-634. ISBN: 978-80-214-4983-1.

[61] HAN, Jiawei, Micheline KAMBER a Jian PEI.: "Data mining: concepts and techniques.", *Morgan Kaufmann series in data management systems.* 2012, ISBN 978-0-12-381479-1.

[62] GARCIA, Vincent, Eric DEBREUVE a Michel BARLAUD. "Fast k Nearest Neighbor Search using GPU,"*In: 2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops.* IEEE, 2008, s. 1-6, ISBN 978-1-4244-2339-2.

[63] LIANG, Shenshen, Ying LIU, Cheng WANG a Liheng JIAN. "A CUDA-based Parallel Implementation of K-Nearest Neighbor Algorithm,"*In: 2009 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery.* IEEE, 2009, s. 291-296, ISBN 978-1-4244-5218-7.

[64] KUANG Q., ZHAO L. "A Practical GPU Based KNN Algorithm,"*IIn Proceedings of the Second Symposium on International Computer Science and Computational Technology (ISCSCT '09) (Dec. 2009).* Academy Publisher, 2009, s. 151-155, ISBN: 978-952-5726-07-7.

[65] GARCIA, Vincent a Frank NIELSEN. "Searching High-Dimensional Neighbours: CPU-Based Tailored Data-Structures Versus GPU-Based Brute-Force Method,"*In Lecture Notes in Computer Science, Springer Berlin Heidelberg* 2009, s. 425-436, ISBN: 978-3-642-01810-7.

[66] PAN, Jia a Dinesh MANOCHA. "Fast GPU-based locality sensitive hashing for k-nearest neighbor computation,"*In: Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '11.* New York, USA: ACM Press, 2011, s. 211-220, ISBN: 978-1-4503-1031-4.

[67] BURGET, R.; UHER, V.; MASEK, J "Image Mining Extension for RapidMiner (Introductory)."*Chapman Hall/CRC Data Mining and Knowledge Discovery Series, Kapitola 20,* ve HOFMANN, M., KLINKENBERG, R., "RapidMiner: Data Mining Use Cases and Business Analytics Applications, "*New York: Taylor & Francis,* 2013, ISBN 1482205491.

[68] KOVACS A., PREKOPCSAK Z.,"Robust GPGPU plugin development for RapidMiner,"*In RapidMiner Community Meeting And Conference - RCOMM 2012,* 2012.

[69] YANAI, K.; KAWANO, Y. "Food image recognition using deep convolutional network with pre-training and fine-tuning."*In: 2015 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)* 2015. s. 1-6. ISBN 978-1-4799-7079-7.

[70] JIA DENG, WEI DONG, R. SOCHER, LI-JIA LI, KAI LI a LI FEI-FEI. "ImageNet: A large-scale hierarchical image database.", *In: 2009 IEEE Conference on Computer Vision and Pattern Recognition* 2009, s. 248-255, ISBN 978-1-4244-3992-8

[71] YING ZHANG; SAIZHENG ZHANG. "Optimized Deep Learning Architectures with Fast Matrix Operation Kernels on Parallel Platform."*In: 2013 IEEE 25th International Conference on Tools with Artificial Intelligence* 2013. s. 71-78. ISBN 978-1-4799-2972-6.

[72] INCHUL SONG; HYUN-JUN KIM ; JEON, P.B. "Deep learning for realtime robust facial expression recognition on a smartphone." *In: 2014 IEEE International Conference on Consumer Electronics (ICCE)* 2014. s. 564-567. ISBN 978-1-4799-1291-9.

[73] HAUSWALD, J.; YIPING KANG; LAURENZANO, M.A.; QUAN CHEN; CHENG LI; MUDGE, T.; DRESLINSKI, R.G.; MARS, J.; LINGJIA TANG. "DjiNN and Tonic: DNN as a service and its implications for future warehouse scale computers."*In: Proceedings of the 42nd Annual International Symposium on Computer Architecture - ISCA '15* New York, New York, USA: ACM Press, 2015, s. 27-40. ISBN 9781450334020.

[74] XIONG LUO, YI CHEN a FENG SHEN. "Text Classification Dimension Reduction Algorithm for Chinese Web Page Based on Deep Learning."*n: International Conference on Cyberspace Technology (CCT 2013)* 2013. s. 451-456. ISBN 978-1-84919-801-1.

[75] IANG, Zhenchao, Lishuang LI, Degen HUANG a LIUKE JIN. "Training word embeddings for deep learning in biomedical text mining tasks. "*In: 2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)* 2015. s. 625-628. ISBN 978-1-4673-6799-8.

[76] PICZAK, Karol J. "Environmental sound classification with convolutional neural networks. " *In: 2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)* 2015. s. 1-6. ISBN 978-1-4673-7454-5.

[77] MCLOUGHLIN, Ian, Haomin ZHANG, Zhipeng XIE, Yan SONG a Wei XIAO. "Robust Sound Event Classification Using Deep Neural Networks. "*IEEE/ACM Transactions on Audio, Speech, and Language Processing* 2015. s. 540-552. ISSN 2329-9290.

[78] A. Krizhevski, I. Sutskever, and G. E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks."*In: NIPS 2012* 2012.

[79] Y. LeCun, B. Boser, J. Denker, et al. "Back-propagation applied to handwritten zip code recognition."*In: Neural Computation* 1989. s. 541-551. ISSN 0899-7667.

[80] YOSHUA BENGIO; IAN J. GOODFELLOW; AARON COURVILLE. "Deep Learning."*Bengio-et-al-2015-Book,* [online]. 2015. Dostupné z URL: `http://www.iro.umontreal.ca/~bengioy/dlbook`

[81] J. Bouvrie. "Notes on convolutional neural networks."*Tech. rep. MIT* 2006.

[82] ARGANDA-CARRERAS, Ignacio, Srinivas C. TURAGA, Daniel R. BERGER, et al. "Crowdsourcing the creation of image segmentation algorithms for connectomics. "*Frontiers in Neuroanatomy* 2015. DOI: 10.3389/fnana.2015.00142.

[83] PRICE, B., MORSE, B., COHEN, S. "Graph Cut Based Image Segmentation with Connectivity Priors."*IEEE CVPR* 2008

[84] VINCENTE, S., KOLMOGOROV, V., ROTHER, C. "Graph Cut Based Image Segmentation with Connectivity Priors."*IEEE CVPR* 2008

[85] SINOP, A., GRADY, L.: "A Seeded Image Segmentation Framework Unifying Graph Cuts and Random Walker which Yields a New Algorithm", *ICCV*, 2007

[86] GRADY, L.: "Random Walks for Image Segmentation", *IEEE Trans. PAMI* 2006, s. 1768–1783

[87] BURGET, R.; RAY, J.,K.;UHER, V.; MAŠEK, J.; DUTTA, M. "Supervised Video Scene Segmentation using Similarity Measures Supervised Video Scene

Segmentation using Similarity Measures." *International Conference on Telecommunications and Signal processing.* 2013. s. 793-797. ISBN: 978-1-4799-0402-0.

[88] MAŠEK, J.; BURGET, R.; UHER, V. "IMMI for Satellite Data Analysis," *In RapidMiner Community Meeting And Conference - RCOMM 2013*, 2013. s. 3-11, ISBN: 978-3-8440-2145-5.

[89] FREUND, Y;SCHAPIRE, R. "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting "*Journal of Computer and System Sciences [online]* s. 119–139, 1997, ISSN: 00220000

[90] GALLANT, S.I. "Perceptron-based learning algorithms."*IEEE Transaction on Neural Networks* s. 179–191, 1990, DOI: 10.1109/72.80230

[91] VIOLA, P.; JONES, M. "Robust Real-Time Face Detection."*International Journal of Computer Vision* s. 137–154, 2004, ISSN: 1573-1405

[92] Khronos OpenCL Working Group, "The OpenCL Specification - v. 2.1,"2015, available: `https://www.khronos.org/registry/cl/specs/opencl-2.1.pdf`

[93] NVIDIA, 2013, February 5. "GeForce Hardware,"available: `http://www.geforce.com/hardware/desktop-gpus/geforce-gtx-690`

[94] OK, AO. "Robust Detection of Buildings from a single color aerial image.", *Proceedings of GEOBIA,*2008,

[95] SIRMACEK, Beril a Cem UNSALAN. "Building detection from aerial images using invariant color features and shadow information.", *23rd International Symposium on Computer and Information Sciences,*2008, s. 1-5, ISBN 978-1-4244-2880-9.

[96] KOMAROV I, DASHTI A, D'SOUZA RM. "Fast k-NNG Construction with GPU-Based Quick Multi-Select.", *PLoS ONE,*2014, DOI: 10.1371/journal.pone.0092409

[97] GAVAHI, Mohsen, Reza MIRZAEI, Abolfazl NAZARBEYGI, Armin AHMADZADEH a Saeid GORGIN. "High performance GPU implementation of k-NN based on Mahalanobis distance.", *Computer Science and Software Engineering (CSSE),*2015, s. 1-6, ISBN 978-1-4673-9181-8.

[98] TANG X., HUANG Z., EYERS D., MILLS S., GUO M. "Efficient Selection Algorithm for Fast k-NN Search on GPUs.", *Parallel and Distributed Processing Symposium (IPDPS),*2015, s. 397-406, DOI: 10.1109/IPDPS.2015.115

[99] MIT cbcl Face Data, "CBCL face database,"dostupné z URL: http://cbcl.mit.edu/cbcl/software-datasets/FaceData2.html

[100] CMU Image Data Base, "Frontal Face Images,"dostupné z URL: http://vasc.ri.cmu.edu//idb/html/face/frontal_images/index.html

[101] TSAI Pin Yi, Yarsun HSU, Ching-Te CHIU a Tsai-Te CHU. "Accelerating AdaBoost algorithm using GPU for multi-object recognition.", *In: 2015 IEEE International Symposium on Circuits and Systems (ISCAS),*2015, s. 738-741, ISBN 978-1-4799-8391-9.

[102] POVODA, L.; BURGET, R.; MAŠEK, J.; DUTTA, M. "Job Shop Scheduling Problem with Heuristic Genetic Programming Operators." *In 2015 2nd International Conference on Signal Processing and Integrated Networks (SPIN).* Noida, Delhi-NCR, India: 2015. s. 702-707. ISBN: 978-1-4799-5990-7.

[103] De Borst GJ, Moll F. "Biology and treatment of recurrent carotid stenosis."*J Cardiovasc Surg (Torino),* 2012, s. 27–34.

[104] JENG, Jiann-Shing a Ping-Keung YIP. "Evaluation of vertebral artery hypoplasia and asymmetry by color-coded duplex ultrasonography."*Ultrasound in Medicine & Biology* 2004, s. 605-609, ISSN 03015629.

[105] BREKKEN, Reidar, Sébastien MULLER, Sjur U. GJERALD a Toril A. Nagelhus HERNES. "Simulation Model for Assessing Quality of Ultrasound Strain Estimation in Abdominal Aortic Aneurysm."*Ultrasound in Medicine & Biology* 2012, s. 889-896, ISSN 03015629.

[106] GOLEMATI, Spyretta, John STOITSIS, Emmanouil G. SIFAKIS, Thomas BALKIZAS a Konstantina S. NIKITA. "Using the Hough transform to segment ultrasound images of longitudinal and transverse sections of the carotid artery."*Ultrasound in Medicine & Biology* . 2007, s. 1918-1932, ISSN 03015629.

[107] RIHA, Kamil a Radek BENES. "Circle detection in pulsative medical video sequence."*In: IEEE 10th INTERNATIONAL CONFERENCE ON SIGNAL PROCESSING PROCEEDINGS [online],* IEEE, 2010, s. 674-677, ISBN 978-1-4244-5897-4.

[108] BENES, Radek, Jan KARASEK, Radim BURGET a Kamil RIHA. "Automatically designed machine vision system for the localization of CCA transverse section in ultrasound images,"*Computer Methods and Programs in Biomedicine.* 2013, s. 92-103. ISSN: 0169-2607.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

α váha lineárního slabého klasifikátoru

A_{CC} Accuracy – přesnost

ALU Arithmetic Logic Unit – aritmeticko-logická jednotka

b bias - odchylka

C počet kategorií

CEITEC Central European Institute of Technology – Středoevropský technologický institut

C_{F} hodnoticí (fitness) funkce

C_{M} Matthewsův korelační koeficient

CNN Convolutional Neural Networks – konvoluční neuronové sítě

CPU central processing unit – počítačový procesor

CT Computed Tomography – počítačová tomografie

CUDA Compute Unified Device Architecture – paralelní programovací architektura vyvinutá společností Nvidia

DBN Deep Belief Networks

DL Deep Learning – hluboké učení

DT Decision Tree – rozhodovací strom

ϵ chyba lineárního slabého klasifikátoru

E střední kvadratická chyba

EA evoluční algoritmy

f feature – příznak

FLOPS FLoating-point Operations Per Second – počet operací typu *float* za sekundu

F_{NR} falešně negativní míra

F_{PR} falešně positivní míra

g výstup neuronové sítě

GPU Graphics Processing Unit – grafický akcelerátor

$\mathbf{H}$ množina všech lineárních slabých klasifikátorů

HOG Histograms of Oriented Gradients – histogramy orientovaných gradientů

$h(x)$ slabý lineární klasifikátor aplikovaný na vstupní snímek x

$H(x)$ silný klasifikátor aplikovaný na vstupní snímek x

i_{I} integral image – integrální obraz

i_{ISqr} kvadrát hodnot integrálního obrazu

JGAP Java Genetic Algorithms Package – Java knihovna genetických algoritmů

k konvoluční filtr

k–NN k–Nearest Neighbor – algoritmus k–nejbližších sousedů

ℓ konvoluční vrstva

LBP local binary pattern – metoda lokárních binárních vzorů

LSH Locality Sensitive Hashing

M množina všech příznakových map

∇f obrazový gradient

$\|\nabla f\|$ modul obrazového gradientu

N_{C} počet všech lineárních klasifikátorů v kaskádě

N_{I} celkový počet snímků

NN Neural Networks – neuronové sítě

N_{TP} počet správně klasifikovaných positivních snímků

OpenCL Open Computing Language – systém pro programování paralelních zařízení

OpenCV Open Source Computer Vision – projekt zabývající se pracováním obrazu

p polarita

P prahová hodnota stupně kaskády

ψ úhel gradientu ∇f

r feature map - výstupní mapa příznaků

RBM Restricted Boltzmann Machines – omezené Boltzmannovy systémy

RF Random Forest – náhodné lesy

RNN Recurrent Neural Networks – rekurentní neuronové sítě

RM RapidMiner

R_{TP} True Positive Rate - míra správných detekcí

R_{FP} False Positive Rate - míra falešně positivních detekcí

SAE Stacked Auto-encoder – auto-enkodéry

SIMT Single Instruction Multiple Threads – architektura jedné instrukce - mnoha vláken

SMs Streaming Multi-processors – streamovací multi-procesory

SoC system on chip – systém na čipu

SVM Support Vector Machine – algoritmus systému podpůrných vektorů

s kumulovaný součet hodnot pixelů v řádku

Θ threshold – prahová hodnota slabého lineárního klasifikátoru

US ultrasound – ultrazvukový

w váha trénovacího snímku x

W matice vah konvolučních vrstev

x vstupní snímek

XML Extensible Markup Language

y klasifikační třída snímku x

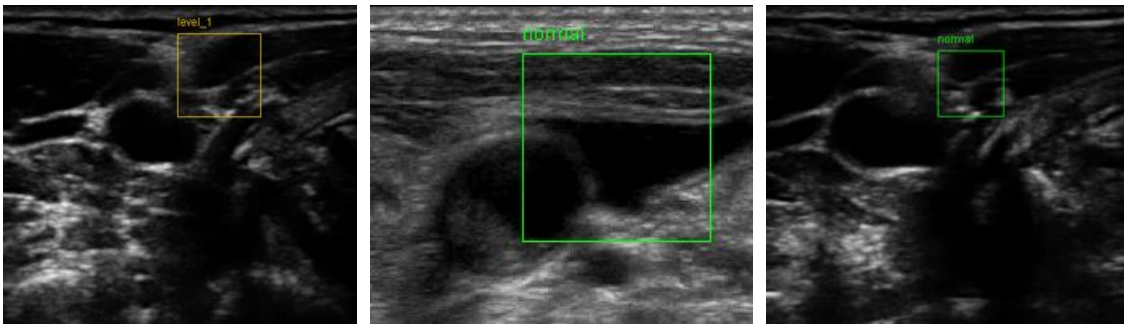
SEZNAM PŘÍLOH

A PŘÍLOHY

A.1 Detekce domů

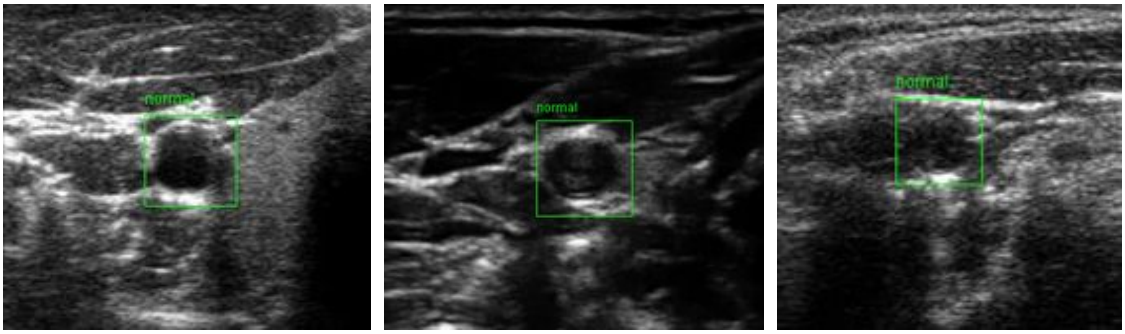
Obr. A.1: Vizualizace výsledků programem Google earth - Lisabon.

A.2 Detekce arterií



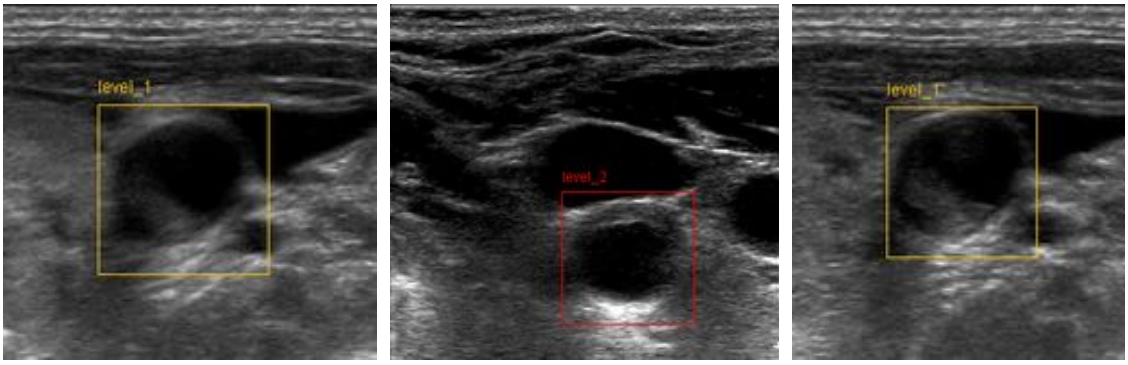
(a) (b) (c)

Obr. A.2: Nesprávně detekované arterie.



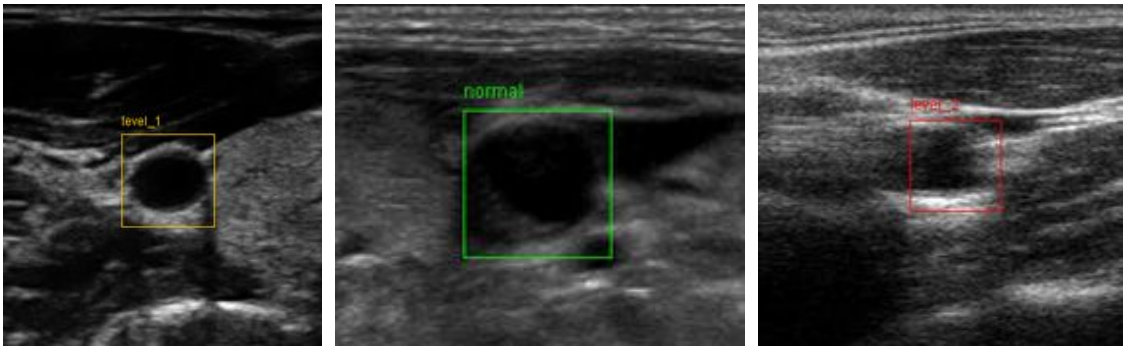
(a) (b) (c)

Obr. A.3: Správné detekce arterií zdravých pacientů.



(a) (b) (c)

Obr. A.4: Správné detekce arterií nemocných pacientů.



(a) Zdravá arterie (b) Nižší stupeň plaku (c) Zdravá arterie

Obr. A.5: Nesprávné určení stupně plaku v artérii.

PUBLIKACE AUTORA

Publikace v časopisech s impaktním faktorem

1. ([2]) POVODA, L.; BURGET, R.; MASEK, J.; UHER, V.; DUTTA, M. K. "Optimization Methods in Emotion Recognition System" *Radioengineering [online],* 2016.
   (IF: 0,59, podíl 10 %)
2. ([1]) ŘÍHA, K.; MAŠEK, J.; BURGET, R.; BENEŠ, R.; ZÁVODNÁ, E. "Novel method for localization of common carotid artery transverse section in ultrasound images using modified Viola–Jones detector." *Ultrasound in Medicine & Biology.* 2013, s. 1887-1902, ISSN 0301-5629.
   (IF: 2,1 podíl 30 %)
3. ([5]) KLUS J., PORIZKA P., MASEK J., RAJNOHA M., BURGET R., PROCHAZKA D., SKARKOVA P., SLADKOVA L., NOVOTNY J., KAISER J. "Multivariate classification of echellograms - new perspective in Laser-Induced Breakdown Spectroscopy analysis.", *Spectrochimica Acta Part B: Atomic Spectroscopy,* 2016, **(odesláno k publikování)**
   (IF: 3,047, podíl 20 %)

Publikace v časopisech bez impaktního faktoru

1. ([3]) MAŠEK, J.; BURGET, R.; POVODA, L.; DUTTA, M. "Multi–GPU Implementation of Machine Learning Algorithm using CUDA and OpenCL." *International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems*, 2016, roč. 5, č. 2, s. 101-107. ISSN: 1805-5443.
   (podíl 69 %)
2. ([56]) POVODA, L.; BURGET, R.; MAŠEK, J.; CVRK, L. "Automatické rozpoznávanie emócií z českého textu pomocou umelej inteligencie." *Elektrorevue - Internetový časopis (http://www.elektrorevue.cz)*, 2015, roč. 17, č. 1, s. 15-18. ISSN: 1213-1539.
   (podíl 10 %)

Publikace v konferenčních sbornících

1. SINGH, A.; RAGHUVANSHI, N.; DUTTA, M.; BURGET, R.; MAŠEK, J. "An SVD Based Zero Watermarking Scheme for Authentication of Medical Images for Tele-medicine Applications." *In Proceedings of the 39th International Conference on Telecommunication and Signal Processing, TSP 2016*, Vídeň, 2016, s 511-514. ISBN: 978-1-5090-1287-9.

(podíl 47 %)

2. ([102]) POVODA, L.; BURGET, R.; MAŠEK, J.; DUTTA, M. "Job Shop Scheduling Problem with Heuristic Genetic Programming Operators." *In 2015 2nd International Conference on Signal Processing and Integrated Networks (SPIN).* Noida, Delhi-NCR, India: 2015. s. 702-707. ISBN: 978-1-4799-5990-7.
(podíl 10 %)

3. ([58]) MAŠEK, J.; BURGET, R.; POVODA, L.; HARVÁNEK, M. "Image Search Using Similarity Measures Based on Circular Sectors." *In Computer Science & Information Technology (CS & IT).* Dubaj, Spojené arabské emiráty: 2015. s. 241-251. ISBN: 978-1-921987-43-4. ISSN: 2231-5403.
(podíl 67 %)

4. ([59]) SINGH, A.; DUTTA, M.; BURGET, R.; MAŠEK, J. "Identification of Acrylamide in Fried Potato Crisps Using Image Processing in Wavelet Domain." *In 2015 38th International Conference on Telecommunications and Signal Processing (TSP).* Prague, Czech Republic: 2015. s. 545-549. ISBN: 978-1-4799-8497-8. ISSN: 1805-5435.
(podíl 47 %)

5. DUTTA, M.; GANGULY, S.; SRIVASTAVA, K.; GANGULY, S.; PARTHASARATHI, M.; BURGET, R.; MAŠEK, J. "An Efficient Grading Algorithm for Non- Proliferative Diabetic Retinopathy using Region Based Detection." *In 2014 37th International Conference on Telecommunications and Signal Processing (TSP).* Berlin, Germany: 2014. s. 503-507. ISBN: 978-80-214-4983-1.
(podíl 42 %)

6. ([60]) KARÁSEK, J.; BURGET, R.; UHER, V.; MAŠEK, J.; DUTTA, M. "Color Image (Dis) Similarity Assessment and Grouping based on Dominant Colors." *In 2014 37th International Conference on Telecommunications and Signal Processing (TSP).* Berlin, Germany: 2014. s. 631-634. ISBN: 978-80-214-4983-1.
(podíl 10 %)

7. ([4]) MAŠEK, J.; BURGET, R.; KARÁSEK, J.; UHER, V.; DUTTA, M. "Multi–GPU Implementation of k-Nearest Neighbor Algorithm,"*In Proceedings of the 38th International Conference on Telecommunication and Signal Processing.* Berlin, Germany: 2014. s. 764-767. ISBN: 978-1-4799-8497-8.
(podíl 54 %)

8. UHER, V.; BURGET, R.; KARÁSEK, J.; MAŠEK, J.; DUTTA, M. "Automatic Image Labelling using Similarity Measures." *In MEDCOM 2014 CD-ROM.* Greater Noida: IEEE, 2014. s. 101-104. ISBN: 978-1-4799-5096-6.

(podíl 20 %)

9. ([87]) BURGET, R.; RAY, J.,K.;UHER, V.; MAŠEK, J.; DUTTA, M. "Supervised Video Scene Segmentation using Similarity Measures Supervised Video Scene Segmentation using Similarity Measures." *In 36th International Conference on Telecommunications and Signal processing.* 2013. s. 793-797. ISBN: 978-1-4799-0402-0.
(podíl 20 %)

10. ([36]) MAŠEK, J.; BURGET, R.; KARÁSEK, J.; UHER, V.; GÜNEY, S. "Evolutionary Improved Object Detector for Ultrasound Images."*In 2013 36th International Conference on Telecommunications and Signal Processing (TSP).* 2013, s. 586-590, ISBN: 978-1-4799-0402-0.
(podíl 49 %)

11. ([37]) MAŠEK, J.; BURGET, R.; UHER, V.; GÜNEY, S. "Speeding up Viola–Jones algorithm using multi–Core GPU implementation."*In 2013 36th International Conference on Telecommunications and Signal Processing (TSP).* 2013, s. 808-812, ISBN: 978-1-4799-0402-0.
(podíl 59 %)

12. ([8]) MAŠEK, J.; BURGET, R.; UHER, V. "IMMI: Interactive Segmentation Toolkit," *In Engineering Applications of Neural Networks. Communications in Computer and Information Science.* Heidelberg: Springer Berlin Heidelberg, 2013. s. 380-387. ISBN: 978-3-642-41012-3, ISSN: 1865-0929.
(podíl 60 %)

13. ([38]) KARÁSEK, J.; BURGET, R.; MAŠEK, J.; BENDA, O. "Genetic Programming Based Classifier in Viola-Jones RapidMiner Image Mining Extension,"*In 2013 36th International Conference on Telecommunications and Signal Processing (TSP).* 2013, s. 872-876. ISBN: 978-1-4799-0402-0.
(podíl 10 %)

14. ([88]) MAŠEK, J.; BURGET, R.; UHER, V. "IMMI for Satellite Data Analysis," *In RapidMiner Community Meeting And Conference - RCOMM 2013*, 2013. s. 3-11, ISBN: 978-3-8440-2145-5.
(podíl 60 %)

15. UHER, V.; BURGET, R.; MAŠEK, J.; DUTTA, M. "3D Brain Tissue Selection and Segmentation from MRI." *In 36th International Conference on Telecommunications and Signal processing.* 2013. s. 839-842. ISBN: 978-1-4799-0402-0.
(podíl 20 %)

16. ([57]) UHER, V.; BURGET, R.; MAŠEK, J.; KARÁSEK, J. "Audio mining extension." *In RapidMiner Community Meeting And Conference - RCOMM 2013.* 2013. s. 12-18. ISBN: 978-3-8440-2145-5.
(podíl 10 %)

17. ([6]) BURGET, R.; UHER, V.; MAŠEK, J. "Trainable Segmentation Based on Local-level and Segment-level Feature Extraction," *In IEEE International Symposium on Biomedical Imaging,* Barcelona: 2012. s. 17-24. ISBN: 978-1-4673-1118-2.
(podíl 15 %)

18. ([39]) MAŠEK, J. "Detekce objektů v obraze s pomocí Haarových příznaků." *In Proceedings of the 18th Conference STUDENT EEICT 2012 Volume 2.* 2012. s. 70-72. ISBN: 978-80-214-4461-4.
(podíl 100 %)

19. ([35]) BURGET, R.; CIKA, P.; ZUKAL, M.; MAŠEK, J. "Automated Localization of Temporomandibular Joint Disc in MRI Images."*In 2011 34th International Conference on Telecommunications and Signal Processing (TSP).* 2011, s. 413-416, ISBN: 978-1-4577-1409-2.
(podíl 20 %)

20. ([9]) KALASOVA D., MASEK J., ZIKMUND T., SPURNY P., BURGET R., KAISER J. "Segmentation of multiphase materials applying trainable segmentation.", *7th Conference on Industrial Computed Tomography, Leuven, Belgium (iCT 2017),* 2017, **(odesláno k publikování)**
(podíl 20 %)

Kapitoly v knize

1. ([67]) BURGET, R.; UHER, V.; MASEK, J "Image Mining Extension for RapidMiner (Introductory)."*Chapman Hall/CRC Data Mining and Knowledge Discovery Series, Kapitola 20,* ve HOFMANN, M., KLINKENBERG, R., "RapidMiner: Data Mining Use Cases and Business Analytics Applications, "*New York: Taylor & Francis,* 2013, ISBN 1482205491.

2. BURGET, R.; UHER, V.; MASEK, J "Image Mining Extension for RapidMiner (Advanced)."*Chapman Hall/CRC Data Mining and Knowledge Discovery Series, Kapitola 21,* ve HOFMANN, M., KLINKENBERG, R., "RapidMiner: Data Mining Use Cases and Business Analytics Applications, "*New York: Taylor & Francis,* 2013, ISBN 1482205491.

Výzkumné zprávy

1. SYSEL, P.; BURGET, R.; MAŠEK, J. "Odborná zpráva o postupu prací za rok 2015: Databázové úložiště."*Brno: VUT v Brně*, 2015.
2. SYSEL, P.; BURGET, R.; MAŠEK, J. "Odborná zpráva o postupu prací za rok 2014: Analýza problematiky vhodného vývojového prostředí."*Brno: VUT v Brně*, 2014.
   (podíl 33 %)
3. SYSEL, P.; BURGET, R.; MAŠEK, J. "Odborná zpráva o postupu prací za rok 2014: Analýza nástrojů a pravidel pro tvorbu dokumentace. "*Brno: VUT v Brně*, 2014.
   (podíl 33 %)
4. SYSEL, P.; BURGET, R.; MAŠEK, J. "Odborná zpráva o postupu prací za rok 2014: Analýza metodiky programování zdrojových kodů."*Brno: VUT v Brně*, 2014.
   (podíl 33 %)
5. SYSEL, P.; BURGET, R.; MAŠEK, J. "Odborná zpráva o postupu prací za rok 2014: Analýza a zprovoznění build systému."*Brno: VUT v Brně*, 2014.
   (podíl 33 %)

Vyvinutý software

1. BURGET, R.; SMÉKAL, Z.; UHER, V.; MAŠEK, J.: Trénovatelná segmentace. *Trainable segmentation of images based on multi-spectral features.* URL: `http://splab.cz/ts`
   (podíl 15 %)
2. BURGET, R.; MAŠEK, J.; UHER, V.; SMÉKAL, Z.: IMMI–OD *Image object detection method for data mining platform with evolutionary optimization.* URL: `http://splab.cz/immi`
   (podíl 25 %)
3. UHER, V.; BURGET, R.; SMÉKAL, Z.; KARÁSEK, J.; MAŠEK, J.: Masker *Image labeling tool.* URL: `http://code.google.com/p/masker/`
   (podíl 10 %)

Curriculum Vitæ

Jan Mašek

Osobní informace

Datum narození:	21. září, 1987
Místo narození:	Trutnov
Telefon:	+420728947018
E-mail:	jan.masek@seznam.cz

Vzdělání

2012–2016	Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Technická 3058/10, 616 00 Brno, obor: Teleinformatika, titul: Ph.D. (předpokládaná doba ukončení: 2016)
2010–2012	Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Technická 3058/10, 616 00 Brno, obor: Telekomunikační a informační technika, titul: Ing.
2007–2010	Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Technická 3058/10, 616 00 Brno, obor: Teleinformatika, titul: Bc.
1999–2007	Gymnázium Trutnov, Jiráskovo náměstí 325, 541 01 Trutnov

Stáže

2013	Philippsova Univerzita v Marburgu, Fakulta matematiky a počítačových věd (Mathematik und Informatik Philipps-Universität Marburg)

Ocenění

2012	Cena děkanky FEKT, VUT v Brně za diplomovou práci

Předchozí zaměstnání

2013–2016	Vědecký pracovník, Centrum senzorických informačních a komunikačních systémů (projekt VaVpI)

Participace na projektech

2016–nyní	Thermostat Data mining (spoluřešitel, 750 669 CZK, hospodářská smlouva, referenční osoba Ondřej Lipták - Leader of electronic design group at Honeywell, Brno Hardware Center of Excellence)
2016–nyní	Projekt segmentace dat meteoritů (spoluřešitel, řešeno s CEITEC)
2016–nyní	Optimalizace návrhu datových skladů (spoluřešitel, 300 000 CZK, podpora výzkumu od 3S s.r.o.)
2016–nyní	Zahájení spolupráce s 3S s.r.o. a Huawei (spoluřešitel, jednání o podpoře výzkumu)
2016–nyní	Klasifikace hutních materiálů pomocí spektrometrie laserem indukovaného plazmatu za použití hlubokých neuronových sítí, IGA reg.č. FEKT/FSI-J-16-3564
2015–nyní	BriskMiner – efektivní nástroj pro pokročilou analytiku podnikových procesů. TAČR reg. č. TH1010277
2015–nyní	SmartMET - Pokročilé meteorologické informace pro letectví (spoluřešitel, projekt s Honeywell International s.r.o.), TAČR reg. č. TA04010625
2014–nyní	Nástroj pro sběr, analýzu a vizualizaci statistických dat o využívání produktů s cílem zvýšit efektivitu vývoje, TAČR reg. č. TA04011024
2014–nyní	Kognitivní multimediální analýza zvukových a obrazových signálů, IGA reg.č. FEKT-S-14-2335
2012–2015	Výzkum a vývoj technologie pro detekci emocí v nestrukturovaných datech, MPO ČR reg.č. FR-TI4/151
2012–2013	Výzkum a vývoj systému pro optimalizaci výrobních procesů, MPO ČR reg.č. FR-TI1/444

Vyžádané přednášky

- House segmentation from aireal images, UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA: La Casita del Estudiante Sede Institucional c/ Juan de Quesada, 30

Vyžádané publikace

- Trainable Segmentation Based on Local-level and Segment-level Feature Extraction, Massachusetts Institute of Technology (MIT), (účast na přípravě)

Publikační aktivita

- Publikace v časopisech s impaktním faktorem: 2 + 1 (odesláno k publikování)
- Publikace v časopisech bez impaktního faktoru: 2
- Publikace v konferenčních sbornících: 19 + 1 (odesláno k publikování)
- Výzkumné zprávy: 5
- Software: 3
- Příspěvky indexované databází WoS: 13
- Příspěvky indexované databází Scopus: 14
- H-index dle databáze WoS: 2
- H-index dle databáze Scopus: 4